ANDRÉ BITTENCOURT LEAL

CONTROLE SUPERVISÓRIO MODULAR DE SISTEMAS HÍBRIDOS

FLORIANÓPOLIS
2005

Universidade Federal de Santa Catarina
Programa de Pós-Graduação em Engenharia Elétrica

CONTROLE SUPERVISÓRIO MODULAR DE SISTEMAS HÍBRIDOS

Tese submetida à
Universidade Federal de Santa Catarina
como parte dos requisitos para a
obtenção do grau de Doutor em Engenharia Elétrica.

ANDRÉ BITTENCOURT LEAL

Florianópolis, Setembro de 2005.

CONTROLE SUPERVISÓRIO MODULAR DE SISTEMAS HÍBRIDOS

André Bittencourt Leal

"Esta Tese foi julgada adequada para a obtenção do título de Doutor em Engenharia Elétrica, Área de Concentração em *Automação e Sistemas*, e aprovada em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Santa Catarina."

José Eduardo Ribeiro Cury, Dr.
Orientador

Alexandre Trofino Neto, Dr.
Coordenador do Programa de Pós-Graduação em Engenharia Elétrica

Banca Examinadora:

José Eduardo Ribeiro Cury, Dr. – UFSC
Presidente

Antonio Eduardo Carrilho da Cunha, Dr. – IME

Antonio Marcus Nogueira Lima, Dr. – UFCG

Guilherme Bittencourt, Dr. – UFSC

Victor Juliano de Negri, Dr. – UFSC

À minha esposa *Simone*.

Aos meus pais *Nédio e Norma*.

"Amor é fogo que arde sem se ver;
É ferida que dói e não se sente;
É um contentamento descontente;
É dor que desatina sem doer;

É um não querer mais que bem querer;
É solitário andar por entre a gente;
É nunca contentar-se de contente;
É cuidar que se ganha em se perder;

É querer estar preso por vontade;
É servir a quem vence, o vencedor;
É ter com quem nos mata lealdade.

Mas como causar pode seu favor
Nos corações humanos amizade,
Se tão contrário a si é o mesmo Amor?"

Luís de Camões.

AGRADECIMENTOS

A *Deus*, pela vida, saúde, família e amigos, sem os quais a realização deste trabalho seria impossível.

Aos meus pais, *Nédio Puccinelli Leal* e *Norma Bittencourt Leal*, pelo amor e carinho com que me criaram e educaram. Eles são os meus orientadores na *Tese da Vida*.

À minha querida esposa *Simone*, que me deu muita força, amor e carinho, elementos indispensáveis nesta etapa tão importante de minha vida.

Ao professor *José Eduardo Ribeiro Cury*, meu orientador, amigo e companheiro nesta jornada. Sob sua orientação foi possível compreender que a concepção de uma tese de doutorado pode ser bem mais que produzir um conhecimento científico. O professor Cury consegue aliar notável conhecimento, seriedade e dedicação ao trabalho, com simplicidade e amizade.

Ao senhores membros da banca examinadora, professores *Antonio Eduardo Carrilho da Cunha*, *Antonio Marcus Nogueira Lima*, *Guilherme Bittencourt*, e *Victor Juliano de Negri*, pela valiosa contribuição que deram para o aperfeiçoamento deste trabalho.

A todos os professores do Departamento de Automação e Sistemas (DAS), em especial aos professores *Edson De Piere* e *Eugênio Castelan*.

Aos amigos *Tatiana Garcia*, *Patrícia Pena*, *César Torrico*, *Max de Queiroz* e *Agnelo Vieira*, alguns dos colegas do grupo de controle de sistemas a eventos discretos e sistemas híbridos, pela amizade e pelas frutíferas discussões sobre o tema de pesquisa.

A todos os amigos que me incentivaram nessa empreitada, em especial àqueles que estiveram mais próximos durante a realização do doutorado: *Cristiane de Oliveira*, *Ricardo Martins* e *Emerson Raposo*. Agradeço também aos amigos conquistados na vida acadêmica no DAS.

À Universidade Federal de Santa Catarina (UFSC), escola que me acolheu para realização dos cursos de graduação, mestrado e doutorado; e ao Programa de Pós Graduação em Engenharia Elétrica da UFSC, em particular agradeço ao Wilson e ao Marcelo.

Aos colegas do Departamento de Engenharia Elétrica da Universidade do Estado de Santa Catarina (UDESC), e à própria UDESC pela oportunidade de realização deste projeto de vida.

À CAPES pelo apoio financeiro dado por intermédio do Programa de Incentivo à Capacitação Docente (PICD).

Resumo da Tese apresentada à UFSC como parte dos requisitos necessários para obtenção do grau de Doutor em Engenharia Elétrica.

CONTROLE SUPERVISÓRIO MODULAR DE SISTEMAS HÍBRIDOS

André Bittencourt Leal

Setembro/2005

Orientador: Prof. José Eduardo Ribeiro Cury, Dr.
Área de Concentração: Automação e Sistemas.
Palavras-chave: Controle Supervisório, Controle Modular, Sistemas Híbridos, Sistemas a Eventos Discretos.
Número de Páginas: 161

A presente tese de doutorado introduz uma metodologia de síntese modular de supervisores para sistemas híbridos que assegura um comportamento não bloqueante e minimamente restritivo para o sistema sob supervisão. Os sistemas híbridos considerados possuem dinâmicas contínuas e dinâmicas a eventos discretos, sendo que a dinâmica discreta é definida por eventos gerados quando variáveis do espaço de estado contínuo alcançam uma superfície de limiar, forçando então transições no estado discreto. A dinâmica contínua é determinada em função do estado discreto atual do sistema. O objetivo de supervisão consiste em restringir a seleção de dinâmicas contínuas de forma que a seqüência de eventos esteja contida em um conjunto de seqüências permitidas (especificação de controle). Os supervisores são projetados (separadamente) usando uma abordagem puramente discreta. Este trabalho contribui ainda para a consolidação da abordagem monolítica de síntese de supervisores para sistemas híbridos e estende resultados anteriores através da consideração de linguagens marcadas. A introdução do conceito de supervisor marcador para sistemas híbridos permite que a determinação do que deve ser uma tarefa fique a cargo do supervisor (da especificação). Nesta tese estuda-se também os problemas que surgem quando o projeto de supervisores (seja na abordagem monolítica ou na abordagem modular) é feito com base em uma aproximação para o comportamento lógico do sistema híbrido. Estabelece-se uma condição sob a qual se pode garantir que um supervisor sintetizado com base em um modelo aproximado para a planta híbrida é também uma solução para o problema original. Exemplos são utilizados para ilustrar a metodologia proposta e provas matemáticas são apresentadas para comprovar os resultados obtidos.

Abstract of Thesis presented to UFSC as a partial fulfillment of the requirements for the degree of Doctor in Electrical Engineering.

MODULAR SUPERVISORY CONTROL OF HYBRID SYSTEMS

André Bittencourt Leal

September 2005

Advisor: José Eduardo Ribeiro Cury, Dr.
Area of Concentration: Automation and Systems
Keywords: Discrete Event Systems, Hybrid Systems, Modular Control, Supervisory Control.
Number of Pages: 161

The present thesis introduces a methodology for the modular synthesis of supervisors for hybrid systems which guarantees a nonblocking and minimally restrictive behavior for the system under supervision. The class of hybrid systems considered is such that threshold-crossing events in the continuous state space force discrete state transitions. The continuous dynamics are determined by a discrete condition, which depends on the current discrete state of the system. The objective of the supervision is to restrict the selection of the continuous dynamics in such a way that the sequence of events be contained in a set of allowed sequences (control specifications). The supervisors are (individually) designed by using a purely discrete framework. This work contributes also to consolidate the monolithic approach to the supervisor synthesis for hybrid systems and extends previous results by considering marked languages. The introduction of a marking supervisor for hybrid systems allows the determination of what is a task in the hybrid system to be in charge of the supervisor (specification). This thesis is also concerned with the problems that arrive when the design of supervisors (in the monolithic or modular approach) is based on an approximated model for the logical behavior of the hybrid systems. It establishes a condition under which one can guarantee that a supervisor synthesized for an approximated model for the hybrid system is also a solution for the original problem. Examples are presented in order to illustrate the proposed methodology and mathematical proofs demonstrate the obtained results.

Sumário

Lista de Figuras

Lista de Abreviaturas

C/E	Condição/Evento
CLP	Controlador de Lógica Programável
e.r.a.	em relação a
PCSE	Problema de Controle Supervisório Equivalente
RW	Ramadge-Wonham
SCEMN	Supervisor C/E Marcador Não Bloqueante
SED	Sistema a Eventos Discretos
SH	Sistema Híbrido
SMN	Supervisor SED Marcador Não Bloqueante
SMSH	Problema de síntese modular (de supervisores) para SHs
SSSH	Problema de síntese de supervisores para SHs

Lista de Símbolos

Conjuntos e funções gerais

$x \in A$	x pertence a A
$x \notin A$	x não pertence a A
$A \subseteq B$	A está contido em B
$A \nsubseteq B$	A não está contido em B
$A \subset B$	A está contido propriamente em B
$A \supseteq B$	A contém B
$A \not\supseteq B$	A não contém B
$A \supset B$	A contém propriamente B
$A \cap B$	intersecção de A com B
$A \cup B$	união de A com B
$A \Rightarrow B$	A implica B
$A \Leftrightarrow B$	A é verdade se e somente se B é verdade
$f : A \to B$	função f de A em B
$f(x)$	valor que a função f toma em $x \in X$

Equações

$\exists$	existe
$\forall$	para todo
:	tal que
$\bigcap\!\!\!\!\bigcirc$	intersecção síncrona de sinais evento
$\bigwedge$	e lógico
t^-	limite pela esquerda ao instante t

Sinais

$x(\cdot)$	trajetória contínua de estados
$x(0)$	valor inicial da trajetória contínua de estados
$y(\cdot)$	sinal contínuo função dos estados, i.e., $y = g(x)$
T	hipersuperfície de limiar
$v(\cdot)$	sinal evento de saída da planta híbrida
$m(\cdot)$	sinal evento de entrada da planta híbrida
$u(\cdot)$	sinal condição de saída da planta híbrida
$w(\cdot)$	sinal condição que registra o valor da trajetória de estados $x(\cdot)$

Conjuntos e espaços de sinais

$\emptyset$ ou $\{\}$	conjunto vazio
X_0	conjunto de estados iniciais
$\mathbb{R}$	conjunto de números reais
2^A	conjunto das partes de A
$\{0,1\}^k$	espaço de vetores de dimensão k cujos elementos assumem valores 0 ou 1
$\{0,1\}^k - \{0\}^k$	espaço de vetores de dimensão k cujos elementos assumem valores 0 ou 1, excluindo-se o vetor nulo (com todos elementos iguais a zero)
V	conjunto finito de eventos de limiar
M	conjunto finito de eventos
U	conjunto finito de condições
V^+	conjunto finito de eventos de limiar, união com o conjunto de eventos de inicialização, isto é, $V^+ = V \cup \{v_0\}$
$V \times U$	produto cartesiano de V e U
$V \times M \times U$	produto cartesiano de V, M e U
$V^+ \times M \times U$	produto cartesiano de V^+, M e U
$\mathcal{V}$	espaço de todos os sinais evento $v(\cdot)$ em $[0, \infty)$
$\mathcal{M}$	espaço de todos os sinais evento $m(\cdot)$ em $[0, \infty)$
$\mathcal{U}$	espaço de todos os sinais condição $u(\cdot)$ em $[0, \infty)$
$\mathcal{V} \times \mathcal{U}$	produto cartesiano de $\mathcal{V}$ e $\mathcal{U}$
$\mathcal{V} \times \mathcal{M} \times \mathcal{U}$	produto cartesiano de $\mathcal{V}$, $\mathcal{M}$ e $\mathcal{U}$
$\mathcal{V} \otimes \mathcal{U}$	produto cartesiano síncrono de $\mathcal{V}$ e $\mathcal{U}$
$\mathcal{V} \otimes \mathcal{M} \otimes \mathcal{U}$	produto cartesiano síncrono de $\mathcal{V}$, $\mathcal{M}$ e $\mathcal{U}$

Modelos e sistemas

$H = (L, \Gamma)$	modelo SED para a planta híbrida $\mathcal{H}$, onde L é a linguagem gerada e Γ é a estrutura de controle
F	supervisor SED
F/H	planta SED sob a ação do supervisor SED (sistema em malha fechada)
$F_1 \wedge F_2$	conjunção dos supervisores F_1 e F_2
$\mathcal{H}_c$	subsistema de dinâmica contínua (subsistema contínuo)
$\mathcal{H}_d$	subsistema de dinâmica discreta (subsistema discreto)
$\mathcal{H}$	sistema híbrido (planta híbrida)
$\mathcal{F}$	supervisor C/E
$\mathcal{F}/\mathcal{H}$	planta C/E sob a ação de supervisor C/E
$\mathcal{F}_1 \wedge \mathcal{F}_2$	conjunção dos supervisores $\mathcal{F}_1$ e $\mathcal{F}_2$
A	aproximação externa para H

Linguagens

ϵ	palavra vazia
v_0	evento de inicialização
$\sigma = vu \in V \times U$	símbolo sobre $V \times U$
V^*	conjunto de todas as palavras de comprimento finito sobre V
$(V \times U)^*$	conjunto de todas as palavras de comprimento finito sobre $V \times U$
$(V^+ \times U)^*$	conjunto de todas as palavras de comprimento finito sobre $V^+ \times U$
$(V^+ \times M \times U)^*$	conjunto de todas as palavras de comprimento finito sobre $V^+ \times M \times U$
$t \in V^*$	palavra em V^*
$s, w \in (V^+ \times U)^*$	palavras sobre $(V^+ \times U)^*$
$s \circ w$	concatenação das palavras s e w
$s + w$	união das palavras s e w
$\mathcal{L}(\mathcal{D})$	linguagem gerada de $\mathcal{D}$
$L, K \subseteq (V^+ \times U)^*$	linguagens sobre $(V^+ \times U)^*$
$E \subseteq V^*$	especificação dada em termos de seqüências de eventos
$K \subseteq (V^+ \times U)^*$	especificação dada em termos de seqüências de pares *evento/condição*
$\overline{K}$	prefixo-fechamento da linguagem K

L^*	fechamento *Kleene* da linguagem L
$L(F/H)$	linguagem gerada em malha fechada
$L_m(F/H)$	linguagem marcada em malha fechada
$P_{_V}:(V^+\times U)^*\rightarrow V^*$	projeção de palavras sobre $(V^+\times U)^*$ em palavras sobre V^*
$P_{_V}(w)$	projeção da palavra $w\in(V^+\times U)^*$ em V^*
$P_V(K)$	projeção da linguagem $K\subseteq(V^+\times U)^*$ em V^*
$P_{_V}^{-1}(E)$	imagem inversa da linguagem $E\subseteq V^*$ pela projeção $P_{_V}$
$V_K(w)$	conjunto ativo de eventos em K após a cadeia $w\in\overline{K}$
$U_K(w,v)$	conjunto ativo de condições em K após a cadeia $w\in\overline{K}$ e para um dado evento $v\in V_K(w)$
$\Sigma_K(w)$	conjunto ativo de condições/eventos em K após a cadeia $w\in\overline{K}$
$\mathbb{C}_{_{VU}}$(K)	classe de linguagens vu-controláveis contidas em K
$\mathbb{C}_{_V}$(E)	classe de linguagens v-controláveis contidas em E
$\mathcal{I}(K_1,K_2)$	conjunto das sublinguagens de K_1 que são interconsistentes e.r.a. K_2
$\mathcal{IC}_{_{VU}}(K_1,K_2)$	conjunto das sublinguagens de K_1 que são vu-controláveis e interconsistentes e.r.a. K_2
$Sup\mathbb{C}_{_{VU}}$(K)	máxima linguagem vu-controlável contida em K
$K^{\uparrow}$	máxima linguagem vu-controlável contida em K
$Sup\mathbb{C}_{_V}$(E)	máxima linguagem v-controlável contida em E
$Sup\mathcal{I}(K_1,K_2)$	máxima sublinguagem de K_1 que é interconsistente e.r.a. K_2
$Sup\mathcal{IC}_{_{VU}}(K_1,K_2)$	máxima sublinguagem de K_1 que é vu-controlável e interconsistente e.r.a. K_2

Outros símbolos

$\diamond$	fim de prova
■	fim de exemplo

Capítulo 1

Introdução

O presente trabalho traz contribuições à teoria de controle supervisório de sistemas híbridos, particularmente à abordagem modular de controle.

Este capítulo está dividido da seguinte forma: na próxima seção apresenta-se o objeto de estudo desta tese e citam-se alguns trabalhos desenvolvidos sobre o tema aqui abordado; na Seção 1.2 enumeram-se as principais contribuições deste trabalho; e, por fim, na Seção 1.3 mostra-se a forma com que este documento está organizado.

1.1 Objeto de Estudo e Trabalhos Relacionados

Os sistemas híbridos constituem uma ampla classe de sistemas dinâmicos formada por sistemas que possuem tanto dinâmicas contínuas quanto dinâmicas discretas. Os formalismos existentes para estes sistemas mostram-se adequados para o tratamento de uma grande variedade de problemas, de forma que os mesmos têm sido considerados nas mais diversas aplicações: na aviação, por exemplo, no controle de tráfego aéreo e nos sistemas de controle de aeronaves; em aplicações automotivas como, por exemplo, os sistemas de controle de tráfego em rodovias e os sistemas de controle de veículos inteligentes; em sistemas de controle de armazenagem e distribuição de petróleo e gás; em sistemas de manufatura; na robótica; na eletrônica de potência; no controle de processos químicos em geral; entre muitas outras.

Tendo em vista que proporcionam um rico contexto para a definição de vários tipos de problemas, nos últimos anos os sistema híbridos foram amplamente estudados por pesquisadores das áreas de computação e de controle. Muitos resultados importantes foram apresentados na literatura, com diferentes enfoques e para as variadas classes de sistemas híbridos.

Na classe de sistemas híbridos considerada neste trabalho, a dinâmica discreta é definida por eventos gerados quando variáveis do espaço de estados contínuo (ou funções destas) alcançam uma superfície de limiar, forçando então transições no estado discreto. A dinâmica contínua, por sua vez, é determinada por uma condição discreta, função do estado discreto atual do sistema. Na literatura, estes sistemas são chamados de sistemas híbridos dirigidos por eventos de limiar *(threshold-event-driven hybrid systems)* (González et al., 2001).

Em termos globais, neste trabalho trata-se do controle supervisório de sistemas híbridos, sendo que o objetivo da supervisão consiste em restringir as transições de estado discreto do sistema híbrido de tal forma que todas as possíveis seqüências de eventos gerados pela planta (sistema híbrido) estejam contidas em um dado conjunto de seqüências de eventos especificado pelo projetista (linguagem desejada para o sistema sob supervisão). Assim, neste trabalho o supervisor é um sistema de dinâmica puramente discreta que observa os sinais da planta híbrida e, em resposta, aplica sinais a esta para restringir o seu comportamento. O problema tratado aqui considera que a planta a ser controlada possui natureza híbrida, podendo ser considerado como uma generalização do problema considerado em (Cury et al., 1998), (Raisch e O'Young, 1998), (Cury e Krogh, 1999), (Koutsoukos et al., 2000) e (Antsaklis e Koutsoukos, 2002), nos quais a natureza híbrida advém da utilização de um supervisor puramente discreto para controlar uma planta contínua. Problemas semelhantes de controle supervisório para sistemas híbridos foram considerados em (Moor et al., 1998), (Moor e Raisch, 1999) e (Moor, Raisch e O'Young, 2002) para sistemas com dinâmicas contínuas em tempo discreto. O problema de controle supervisório de sistemas híbridos considerando a natureza híbrida da planta foi estudado também em (González, 2000) e (González et al., 2001).

Nesta tese, utiliza-se o mesmo paradigma de modelagem empregado em (González et al., 2001) e uma série de resultados obtidos naquele trabalho são considerados aqui. Muitos deles, entretanto, tiveram que ser adaptados e outros foram estendidos para o contexto deste trabalho.

Entretanto, o enfoque específico desta tese consiste na abordagem modular de controle supervisório de sistemas híbridos. Nesta abordagem projeta-se um controlador individual para atender cada um dos componentes da especificação global, e os supervisores modulares são implementados de forma concorrente no intuito de que a ação conjunta destes garanta o cumprimento da especificação global. Além deste trabalho, a síntese modular de supervisores para sistemas híbridos foi abordada apenas em (Moor, Davoren e Raisch, 2001). Entende-se então

que o *estado da arte* neste tema consiste na união das contribuições obtidas nestes dois trabalhos. Existem diferenças significativas entre estes, a começar pela abordagem empregada na modelagem do sistema híbrido. Em (Moor, Davoren e Raisch, 2001) o problema é formulado usando a teoria comportamental de sistemas (Willems, 1991), enquanto que aqui a formulação é feita por intermédio da teoria de sistemas condição/evento (Sreenivas e Krogh, 1991). Por sua vez, a maior contribuição deste trabalho é a garantia de sempre se obterem supervisores modulares que levam a uma solução ótima. Estas e outras diferenças existentes entre o presente trabalho e o de Moor, Davoren e Raisch (2001) são discutidas com mais propriedade no Capítulo 4.

Na abordagem proposta neste trabalho, para a realização da síntese de supervisores, seja na abordagem monolítica, seja na modular, deve-se obter um modelo discreto que represente o comportamento lógico da planta híbrida. Este comportamento é dado pelas seqüências de eventos e dinâmicas contínuas adotadas nos instantes de ocorrência destes eventos, conforme será visto em detalhes no Capítulo 3. A obtenção de modelos lógicos para os sistemas híbridos é, entretanto, um dos maiores problemas relacionados ao controle supervisório destes sistemas. Segundo Chase et al. (1993), em geral, um modelo com número finito de estados para um sistema híbrido não existe ou é computacionalmente difícil de ser obtido. Cury et al. (1998) propõem a obtenção de um modelo de estados finitos cujo comportamento lógico contém o comportamento seqüencial exato da planta híbrida. Todavia, a utilização deste modelo aproximado gera alguns problemas, conforme discutido no Capítulo 6. Deve-se ressaltar que o estudo de metodologias para a obtenção destas aproximações não será contemplado no contexto deste trabalho. Considera-se então que o modelo lógico (aproximado ou exato) da planta híbrida é fornecido ou é obtido por intermédio de uma ferramenta computacional. Assim como em (González, 2000) e (González et al., 2001), utilizou-se uma ferramenta de verificação para obtenção de um modelo aproximado para o sistema híbrido.

Na seqüência, apresenta-se um sumário das principais contribuições desta tese.

1.2 Resumo das Contribuições

De forma bastante sucinta, pode-se dizer que as principais contribuições desta tese são:

- a consolidação da abordagem monolítica para a síntese de supervisores para uma classe de sistemas híbridos;

- a extensão da teoria de controle supervisório de sistemas híbridos para linguagens marcadas;
- a proposição de uma metodologia de síntese modular de supervisores para sistemas híbridos que garante a otimalidade da solução; e
- o estabelecimento de condições sob as quais se pode garantir que supervisores sintetizados com base em um modelo aproximado para a planta híbrida são uma solução também para a planta real (híbrida).

Estas contribuições foram apresentadas à comunidade científica por intermédio das seguintes publicações: (Leal e Cury, 2002), (Leal e Cury, 2004a), (Leal e Cury, 2004b), (Leal e Cury, 2004c) e (Leal e Cury, 2005).

Além disso, durante a realização deste trabalho, participou-se ativamente no desenvolvimento de algoritmos para a síntese modular de supervisores para sistemas híbridos, algoritmos estes que foram desenvolvidos e implementados no contexto da dissertação de mestrado de Rodrigues (2004). Desta forma, entende-se que a ferramenta obtida com a implementação destes algoritmos consiste numa contribuição indireta do presente trabalho.

1.3 Organização do Documento

O restante deste documento está organizado conforme segue:

No Capítulo 2 apresenta-se uma introdução aos sistemas híbridos, fazendo um breve histórico da evolução de seu estudo ao longo das últimas décadas e discute-se sobre algumas das abordagens existentes nas perspectivas das comunidades de controle e de computação. Por fim, enumeram-se alguns exemplos de aplicação de sistemas híbridos e descrevem-se brevemente algumas ferramentas que podem ser utilizadas para a modelagem, simulação, verificação e/ou síntese de controladores para este tipo de sistema.

No Capítulo 3 é feita a formulação de um problema de controle supervisório para sistemas híbridos, apresentando-se a classe de sistemas híbridos tratada ao longo de todo o documento. Tal formulação é feita por intermédio do formalismo de sistemas condição/evento (Sreenivas e Krogh, 1991), mas a resolução do problema proposto é obtida pela resolução de um problema equivalente apresentado no domínio de sistemas a eventos discretos (SEDs). Um exemplo simples é utilizado de forma a ilustrar a metodologia de resolução apresentada.

No Capítulo 4 trata-se do tema central deste trabalho, o controle supervisório modular de sistemas híbridos. Da mesma forma que no Capítulo 3, formula-se o problema de controle por intermédio do paradigma de sistemas condição/evento, mas sua resolução é feita no domínio de SEDs. Apresenta-se então uma metodologia para a solução de problemas de síntese de supervisores modulares para sistemas híbridos e estabelecem-se condições sob as quais tal metodologia leva a uma solução ótima, ou seja, condições sob as quais se obtém supervisores cuja ação conjunta é não bloqueante e minimamente restritiva. O exemplo utilizado no Capítulo 3 é retomado para a resolução por intermédio da abordagem modular, permitindo então que os resultados sejam comparados. Apresenta-se ainda um novo problema, baseado num sistema de controle de trens em uma malha ferroviária, e soluciona-se o mesmo utilizando a abordagem modular proposta.

No Capítulo 5 discute-se sobre a existência de soluções ótimas para a abordagem modular. Estabelece-se então uma metodologia para a síntese modular que sempre leva a uma solução ótima. Assim, se um dado problema de controle supervisório definido para um sistema híbrido tem solução por intermédio da abordagem monolítica, certamente o mesmo resultado é obtido por intermédio da metodologia de síntese modular proposta neste capítulo.

No Capítulo 6 trata-se dos problemas que surgem quando a síntese de supervisores é feita com base em um modelo aproximado para o comportamento lógico do sistema híbrido. Estabelecem-se então condições sob as quais se pode garantir que os supervisores sintetizados com base em um modelo aproximado para a planta híbrida são também uma solução para o problema original, quer seja na abordagem monolítica, quer seja na abordagem modular.

Finalmente, no Capítulo 7 apresentam-se as principais conclusões deste trabalho e as perspectivas de extensão dos resultados obtidos nesta tese.

Capítulo 2

Sistemas Híbridos

2.1 Introdução

Um sistema é dito ser *contínuo no tempo* se seus estados variam continuamente com o tempo, ou seja, se suas variáveis são funções contínuas do tempo. De um modo geral, os sistemas contínuos têm suas variáveis medidas continuamente no tempo, podendo então ser modelados por equações diferenciais. Entretanto, um sistema de natureza contínua pode ter suas variáveis medidas apenas em instantes discretos no tempo, por intermédio de amostragens, dando origem aos chamados sistemas a dados amostrados ou sistemas de tempo discreto, os quais podem ser modelados por equações a diferenças. Em ambos os casos, a variável tempo (t no caso contínuo e k no caso discreto) é naturalmente uma variável independente que aparece como o argumento das funções entrada, estado e saída.

Outros sistemas, estes de natureza bastante diferente dos sistemas contínuos, são os chamados *Sistemas a Eventos Discretos (SEDs)*. Antes de introduzir uma definição para estes sistemas, vamos apresentar o conceito de evento. Um evento pode ser identificado com uma ação (aperto de uma tecla), uma ocorrência espontânea (pane em uma máquina) ou o resultado de várias condições satisfeitas em um dado instante (um equipamento termina uma operação). Estes eventos não dependem diretamente da variável tempo e podem ocorrer a qualquer instante. Um sistema a eventos discretos (SED) é um sistema a estado discreto dirigido por eventos, ou seja, um SED é um sistema em que o estado assume valores num conjunto discreto enumerável, e cuja evolução depende inteiramente da ocorrência de eventos discretos assíncronos no tempo. Embora o termo SED seja muito semelhante e ocasione confusões destes com os sistemas de tempo discreto (a dados amostrados), tais sistemas são de naturezas bastante dis-

tintas. Note que nos sistemas a eventos discretos os estados só podem mudar de um valor para outro quando ocorrem eventos, independentemente do instante em que tais eventos ocorrem. Por outro lado, em um sistema a dados amostrados os estados variam em intervalos de tempo predeterminados, ou seja, nos instantes nos quais são colhidas as amostras, independendo da ação de eventos.

Uma terceira classe consiste nos chamados *sistemas híbridos*. Genericamente, o termo híbrido refere-se à combinação de dois tipos de objetos ou metodologias fundamentalmente diferentes. Segundo Pettersson e Lennartson (1996), um sistema que combina redes neurais artificiais com lógica nebulosa pode ser visto como híbrido (Jou et al., 1999). Um sistema modelado pela interconexão entre sistemas com parâmetros concentrados e parâmetros distribuídos também pode ser visto como um sistema híbrido. Na área de controle, um tipo de sistema híbrido muito conhecido é o sistema de controle digital de processos contínuos. Nestes, uma planta contínua, descrita por equações diferenciais, é controlada por um controlador de tempo discreto, descrito por equações a diferenças. Tais sistemas são híbridos no sentido de combinarem sinais de tempo contínuo com sinais de tempo discreto (Lemmon et al., 1999).

Os sistemas híbridos aos quais nos referimos neste trabalho são sistemas que possuem tanto dinâmicas contínuas quanto dinâmicas de eventos discretos, sendo que, além de coexistirem, estas também interagem entre si. Não apenas o sistema pode ter uma característica híbrida, mas a especificação sobre o comportamento desejado para o mesmo também pode ser híbrida.

No passado, os sistemas híbridos eram freqüentemente abordados de forma que as dinâmicas dirigidas a eventos eram estudadas separadamente das dinâmicas dirigidas pelo tempo. As técnicas disponíveis para a modelagem, análise e síntese de controladores, bem como as ferramentas associadas a estas técnicas, não eram capazes de tratar conjuntamente tais dinâmicas. As primeiras eram geralmente modeladas por autômatos ou redes de Petri e as últimas por equações diferenciais ou a diferenças. Assim, sistemas de natureza híbrida eram convertidos em entidades puramente contínuas ou puramente discretas, de acordo com o objetivo a ser alcançado (Nerode e Kohn, 1993). Entretanto, na maioria dos casos não triviais estas duas dinâmicas interagem de forma significativa, tal que não podem ser desacopladas efetivamente por nenhum tipo de abstração e precisam ser analisadas simultaneamente. Assim, para entender com profundidade o comportamento de um sistema e/ou para alcançar especificações de alto desempenho, em geral torna-se necessário modelar e analisar conjuntamente tais dinâmicas, bem como suas interações.

Tendo em vista que na maioria das vezes as técnicas e ferramentas existentes se mostravam inadequadas para o tratamento de sistemas híbridos, pesquisadores ligados à teoria de controle e à ciência da computação foram levados a repensar e reelaborar conceitos, modelos e técnicas existentes de forma a serem aplicados aos sistemas híbridos. Sendo assim, conceitos básicos tais como os de observabilidade, controlabilidade e estabilidade foram revistos pela comunidade de controle, assim como técnicas de verificação de programas foram revistas pelos pesquisadores da computação, por exemplo.

O restante do capítulo está organizado como segue. Na próxima seção é feita uma breve revisão histórica sobre os sistemas híbridos, mostrando a evolução de seu estudo ao longo de mais de 50 anos. Na Seção 2.3 são apresentadas as principais abordagens existentes para o tratamento de problemas relacionados aos sistemas híbridos sob as perspectivas das comunidades de controle e de ciência da computação, comunidades estas que têm contribuído fortemente para o desenvolvimento de resultados sobre o tema. Na Seção 2.5 são citados alguns exemplos de aplicação de sistemas híbridos. Por fim, na Seção 2.6 são apresentadas as principais conclusões do estudo feito neste capítulo.

2.2 Breve Histórico

Muito embora o estudo de sistemas híbridos tenha se tornado popular apenas recentemente, vários tipos de sistemas que recaem nesta categoria já foram estudados anteriormente. Sistemas dinâmicos descontínuos foram objeto de estudo sistemático na antiga União Soviética e em outros países do leste europeu por um longo período, começando no final dos anos 40. Muito desta teoria foi desenvolvida com estreita ligação com a teoria de controle, a qual proporcionou diversos exemplos motivacionais, tais como controles com relés e controle “bang-bang”. Exposições sistemáticas de resultados desta pesquisa estão disponíveis em vários livros, dentre os quais pode-se citar os de Andronov e Chaikin (1949), Tsypkin (1984) e Filippov (1988). Mais tarde, no final dos anos 60, alguns pesquisadores tais como Kalman et al. (1969) discutiram metodologias matemáticas para estudar sistemas com dinâmicas contínuas e discretas.

Além do controle com relés e do controle “bang-bang”, outros campos importantes de estudos ancestrais de controle híbrido são o controle com estrutura variável (Utkin, 1992), o controle por modos deslizantes (Utkin, 1977; Young e Özgüner, 1999) e o controle digital (Franklin et al., 1990; Kuo, 1992; Ästrom e Wittenmark, 1997).

A primeira referência direta que se conhece sobre sistemas híbridos é o trabalho visionário de Witsenhausen (1966). Nele, o autor trata um problema de controle ótimo para uma classe de sistemas híbridos com dinâmica de tempo contínuo. Este trabalho foi seguido por Pavlidis (1967), o qual estudou a estabilidade de sistemas com impulsos utilizando funções de Lyapunov, e por Rozenvasser (1967), o qual apresentou equações gerais de sensibilidade com relação a um parâmetro para sistemas descontínuos de Equações Diferenciais Ordinárias (EDOs). Fahrland (1970) foi o primeiro autor a defender o desenvolvimento de linguagens de simulação para sistemas que combinam dinâmicas contínuas com dinâmicas de eventos discretos. A incorporação de dinâmicas discretas na simulação de sistemas contínuos foi iniciada por Cellier (1979), o qual foi um dos primeiros a introduzir um conceito de estruturação para sistemas híbridos. A seguir, Johnson (1981) abordou modelos analíticos de processos com múltiplos estágios e Thompson (1982) estudou a utilização de setores cônicos para a análise de sistemas híbridos. Baseado na teoria de autômatos, Wimpey (1982) utilizou controladores de estados finitos para processos contínuos de tempo discreto.

Segundo Antsaklis (2000a), a partir do final dos anos 80 houve uma renovação do interesse pelo tema junto à comunidade de sistemas de controle. Tal renovação foi motivada, em parte, *(i)* pelo desenvolvimento da teoria de controle de SEDs, que ocorreu no final dos anos 80; *(ii)* pelo surgimento do controle adaptativo, nos anos 70 e 80; e *(iii)* pelo interesse renovado em formulações do controle ótimo.

Motivado pelo problema de modelagem de sistemas com histerese, Tavernini (1987) utilizou autômatos diferenciais para modelar sistemas híbridos e apresentou soluções para problemas com valor inicial e suas aproximações numéricas. Ezzine e Haddad (1989) examinaram estabilidade, controlabilidade e observabilidade de uma classe restrita de sistemas híbridos, os sistemas lineares chaveados.

Göllü e Varaiya (1989) utilizaram como exemplo um controlador de disco de computador (*disk drive*) para estudar a consistência de uma estrutura de controle com múltiplas camadas. Neste trabalho os autores utilizaram uma interface para acoplar as partes contínua e discreta do sistema híbrido. Peleties e DeCarlo (1988) discutiram um exemplo de utilização de um recurso pesqueiro com o intuito de demonstrar o modelo proposto para sistemas híbridos. Neste modelo, uma rede de Petri foi utilizada para representar os aspectos de eventos discretos (gerenciamento) do sistema híbrido, enquanto os aspectos de tempo contínuo (produção e colheita) foram capturados por uma equação diferencial não linear de primeira ordem. Logo

depois, os mesmos autores apresentaram outro trabalho semelhante, onde uma rede de Petri modelava o sistema de tomada de decisão, um modelo baseado em equações diferenciais representava o sistema de tempo contínuo e uma interface transmitia informações entre os dois componentes do sistema híbrido (Peleties e DeCarlo, 1989).

Benveniste e Guernic (1990) introduziram a linguagem SIGNAL para especificação e controle de sistemas dinâmicos híbridos. Ramadge (1990) e Chase et al. (1993) analisaram de forma qualitativa os tipos de comportamentos que podem ser exibidos quando dinâmicas contínuas e discretas interagem.

Em paralelo houve um crescente interesse pelos sistemas híbridos entre cientistas da computação e estudiosos de lógica. O advento de computadores digitais fez com que os sistemas híbridos se tornassem muito comuns, pois sempre que um dispositivo digital interage com o mundo contínuo, o comportamento envolve fenômenos híbridos que precisam ser analisados e compreendidos. De fato, sempre que o comportamento de um programa de computador depende de valores de variáveis contínuas, precisa-se de metodologias de sistemas híbridos para garantir a correção do programa. Surgiram então alguns trabalhos sobre verificação de sistemas de tempo real e de sistemas reativos (Joseph, 1988; Alur et al., 1990; Benveniste e Berry, 1991; Ernberg et al., 1991; Manna e Pnueli, 1991; de Bakker et al., 1992). Na mesma época, Maler et al. (1992) introduziram os diagramas de estados híbridos *(hybrid statecharts)* para a especificação de comportamentos híbridos, e ilustraram a metodologia por intermédio do exemplo do "gato e rato". Esta mesma abordagem foi utilizada posteriormente por Manna e Pnueli (1993).

Os primeiros encontros sobre sistemas híbridos ocorreram no início dos anos 90. Organizado por Robert L. Grossman e Anil Nerode, o primeiro *workshop* sobre sistemas híbridos foi realizado junto ao Instituto de Ciências Matemáticas da Universidade de Cornell, nos Estados Unidos, em junho de 1991. Pouco mais de um ano depois, em outubro de 1992, na Universidade Técnica de Lyngby, na Dinamarca, foi realizado o segundo *workshop* sobre sistemas híbridos. Inspirado nestes dois eventos, em 1993 foi editado o primeiro volume sobre sistemas híbridos da série *Lecture Notes in Computer Science*, intitulado *Hybrid Systems* (Grossman et al., 1993). A partir daí o tema despertou grande interesse junto à comunidade científica e então surgiram muitos trabalhos importantes. Na seqüência vieram os títulos *Hybrid Systems II* (Antsaklis et al., 1995), *Hybrid Systems III: Verification and Control* (Alur, Henzinger e Sontag, 1996), *Hybrid Systems IV* (Antsaklis et al., 1997) e *Hybrid Systems V* (Antsaklis et al., 1999), to-

dos resultantes de *workshops*. Nesse intermédio, em março de 1997, na cidade de Grenoble – França, ocorreu o evento denominado *Hybrid and Real-Time Systems* (Maler, 1997). Em abril de 1998, na Universidade de Berkeley, na Califórnia – USA, ocorreu o primeiro de uma série de *workshops* internacionais sobre sistemas híbridos intitulados *Hybrid Systems: Computation and Control (HSCC)* (Henzinger e Sastry, 1998). Outras conferências da série *HSCC* foram realizadas posteriormente nos seguintes locais e datas: em Nijmegen – Países Baixos, em 1999 (Vaandrager e van Schuppen, 1999), em Pittsburgh, Pennsylvania – USA, em 2000 (Lynch e Krogh, 2000), em Roma – Itália, em 2001 (Benedetto e Sangiovanni-Vincentelli, 2001), em Stanford, California – USA, em 2002 (Tomlin e Greenstreet, 2002), em Praga – República Checa, em 2003 (Wiedijk et al., 2003), na Philadelphia, Pennsylvania – USA, em 2004 (Alur e Pappas, 2004) e em Zurich – Suíça, em 2005 (Morari e Thiele, 2005). Em 2006 a conferência será realizada em Santa Barbara, California – nos Estados Unidos.

Existe ainda uma conferência organizada pela *IFAC – International Federation on Automatic Control*, denominada *IFAC conference on Analysis and Design of Hybrid Systems (ADHS)*. A primeira conferência da série foi realizada em 2003 em Saint Malo – França (Engell, 2003) e a próxima será realizada em junho de 2006 na cidade de Alghero – Itália.

Outras publicações deram especial atenção para problemas relacionados a sistemas híbridos, dentre as quais pode-se citar (Pnueli e Sifakis, 1995), (Antsaklis e Lemmon, 1998b), (Antsaklis e Nerode, 1998b), (Evans e Savkin, 1999), (Morse et al., 1999b), (Antsaklis, 2000b), (Lygeros et al., 2002) e (Lygeros, 2004), entre outras.

Existem ainda várias teses de doutorado que abordam o tema sistemas híbridos, tais como os trabalhos de Cellier (1979), Thompson (1982), Barton (1992), Peleties (1992), Andersson (1994), Deshpande (1994), Branicky (1995), Puri (1995), Lygeros (1996), Livadas (1997), Mosterman (1997), Davoren (1998), Tomlin (1998), Chutinan (1999), Fábián (1999), Pettersson (1999), Sipma (1999) e Zad (1999). Mais recentemente, Dang (2000), González (2000), Koo (2000), Djenini (2001), Tabuada (2001), Fehnker (2002), Palomino Bean (2002), Hedlund (2003), Ivancic (2003), Miranda (2003), Villani (2003), da Silva Jr. (2004), Juloski (2004), Girard (2004), Shaikh (2004), Thomas (2004), Corona (2005), Frehse (2005a), e Julius (2005) publicaram suas teses sobre sistemas híbridos.

Recentemente, van der Schaft e Schumacher (2000) lançaram um livro intitulado *"An Introduction to Hybrid Dynamical Systems"*, que, conforme o nome sugere, se trata de um livro introdutório sobre sistemas híbridos. Nele os autores tratam da modelagem, análise e

controle de sistemas híbridos e mostram vários exemplos deste tipo de sistema.

Para uma boa revisão sobre o tema sugere-se a leitura dos artigos estendidos *(surveys)* publicados por Labinaz et al. (1997), Antsaklis et al. (1998), Krogh (2000), Antsaklis e Koutsoukos (2003) e Krogh (2002), entre outros. Para uma revisão mais rápida e superficial consultar os trabalhos de Antsaklis e Lemmon (1998a), Antsaklis e Nerode (1998a), Morse et al. (1999a) e Antsaklis (2000a).

2.3 Abordagens para Sistemas Híbridos

Tendo em vista a ampla área de abrangência dos sistemas híbridos, muitas foram as abordagens e os enfoques dados ao seu estudo nos temas modelagem, análise, síntese e simulação. Em termos amplos, as abordagens diferem em relação a *(i)* se é dada maior ênfase às dinâmicas contínuas ou às dinâmicas discretas, e *(ii)* se a busca é por resultados de análise e síntese ou se o interesse é apenas na análise ou na simulação destes sistemas. Em um extremo do espectro existem abordagens que buscam estender a teoria clássica de sistemas de tempo contínuo (descritos por equações diferenciais ordinárias) de forma a incluir variáveis de tempo discreto ou variáveis que exibem saltos, ou ainda estender resultados para sistemas com chaveamentos. Tipicamente estas abordagens são eficientes para tratar com dinâmicas contínuas complexas e enfatizam resultados sobre estabilidade. No outro extremo do espectro estão as abordagens baseadas em modelos e métodos provenientes da ciência da computação e que visam estender as metodologias de verificação de SEDs para o contexto de sistemas híbridos. Tipicamente, estas abordagens são capazes de tratar de dinâmicas discretas complexas, descritas por autômatos de estados finitos, e enfatizam resultados de análise (verificação) e metodologias de simulação. Ao longo do espectro acima citado, existem ainda muitas outras metodologias que combinam conceitos de sistemas de controle contínuo com conceitos da teoria de controle supervisório de SEDs de forma a obter resultados de análise e de síntese para sistemas híbridos.

A seguir, dividem-se as diferentes abordagens nas perspectivas das comunidades de controle e de ciência da computação.

2.3.1 Perspectiva da Teoria de Controle

Na comunidade de controle, abordagens de modelagem, análise e projeto de controladores para sistemas híbridos estenderam a teoria de sistemas dinâmicos para incluir modos discretos

de operação. Assim, sistemas chaveados (Morse, 1997) e sistemas com dinâmicas descontínuas (Filippov, 1988), por exemplo, podem ser tratados como sistemas híbridos com estruturas especiais.

Na verdade, pode-se entender o *controle com estrutura variável*, *controle por modos deslizantes*, *controle por relés*, *controle por chaveamento de ganhos* e o *controle com lógica nebulosa* como exemplos de sistemas de controle híbrido. A característica comum a todos estes esquemas de controle é sua natureza de chaveamento; com base na evolução da planta (sistema de tempo contínuo a ser controlado) e/ou no progresso do tempo, o supervisor chaveia de um regime de controle para outro (van der Schaft e Schumacher, 2000, p. 140).

A estrutura mais comumente encontrada dentre a classe de sistemas chaveados é a mostrada na Figura 2.1. Neste caso, baseado nos sinais de entrada e saída da planta, o supervisor deve decidir qual dos controladores deve ser usado.

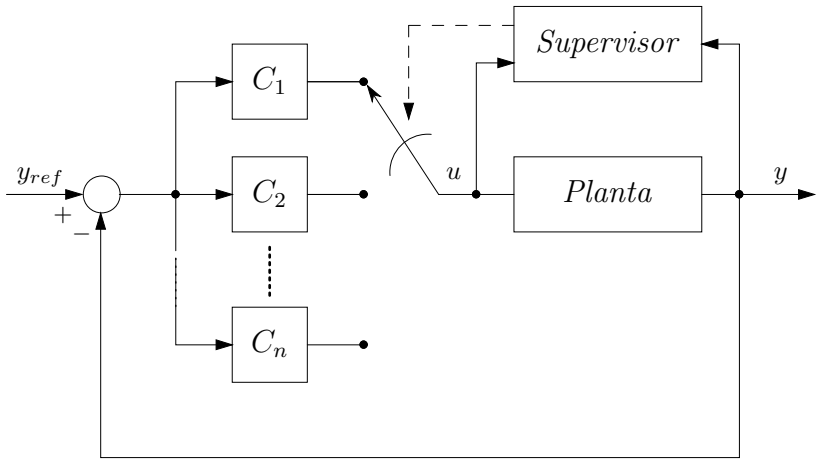


Figura 2.1: Estrutura de controle para sistemas chaveados.

Tendo em vista que esta estrutura tem sido utilizada com sucesso na prática há muitas décadas, vários pesquisadores voltaram a atenção para a sua modelagem e análise. Desde então, um importante tema de pesquisa junto à comunidade de controle tem sido a extensão de definições e condições para a estabilidade de sistemas contínuos para o contexto de sistemas híbridos (Branicky, 1998; Ye et al., 1998; Liberzon e Morse, 1999; Michel, 1999; Michel e Hu, 1999; DeCarlo et al., 2000; Davrazos e Koussoulas, 2001; Pettersson e Lennartson, 2003; Hespanha, 2004) e sistemas chaveados (Fang et al., 2004; Xie e Wang, 2005). Com o intuito de aplicar os resultados teóricos sobre estabilidade de sistemas híbridos, alguns pesquisadores utilizam conceitos de estabilidade de Lyapunov combinados com métodos compu-

tacionais baseados em desigualdades lineares matriciais – LMIs, ver por exemplo os trabalhos de Johansson (1999), Pettersson (1999), Li et al. (2000), Palomino Bean (2002), Pettersson e Lennartson (2002) e de Xie et al. (2003). Outros pesquisadores (Liberzon, 2000) obtiveram resultados importantes sobre a estabilidade de sistemas não lineares utilizando realimentação descontínua.

Outro importante tema de pesquisa junto à comunidade de controle tem sido o *controle supervisório de sistemas híbridos*. Inspirados na estrutura utilizada para o controle digital, alguns autores propuseram uma estrutura análoga para o controle supervisório de sistemas híbridos, na qual a comunicação entre a planta e o supervisor é feita por uma interface formada por conversores A/D e D/A generalizados, conforme ilustrado na Figura 2.2.

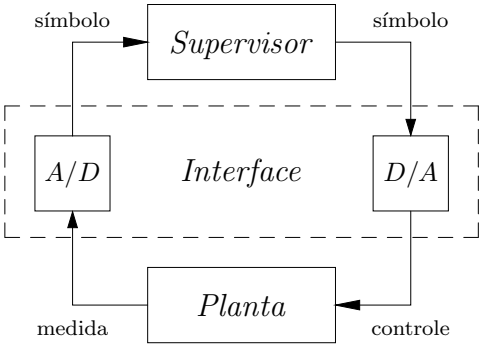


Figura 2.2: Estrutura genérica para o controle supervisório.

Nesta estrutura, a planta evolui ao longo do tempo até que uma variável de seu espaço de estados contínuo cruza determinado limite e então a interface sinaliza a ocorrência de um evento para o supervisor. Este, por sua vez, atualiza seu estado discreto e, de acordo com o estado atual, envia um sinal (também discreto) para a interface, a qual transforma este em um sinal contínuo a ser aplicado na planta.

Na metodologia de controle digital, pode-se realizar todo o projeto do controlador no domínio do tempo, e então aproximar ou emular o controlador por um equivalente discreto. Uma outra alternativa consiste em obter primeiro um *modelo discreto* da planta tomada em conjunto com a interface, para então realizar o projeto do controlador no domínio discreto. A analogia feita no contexto de controle supervisório de sistemas híbridos consiste em, ao invés de obter um modelo discreto, obter um modelo de *eventos discretos* da planta em conjunto com a interface (ver Figura 2.3), utilizando então autômatos ou redes de Petri. Assim, na perspectiva do supervisor, o conjunto planta-interface é um sistema a eventos discretos, pois

recebe e envia apenas sinais discretos e então o controlador é projetado usando metodologias de controle supervisório de SEDs.

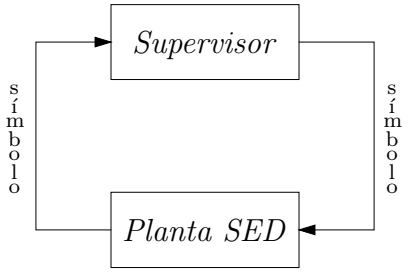


Figura 2.3: Comportamento do conjunto planta-interface como um SED.

Nesta abordagem, a ênfase é dada na ordenação lógica das seqüências de eventos da planta. Assim, os problemas tratados na abordagem de controle supervisório de sistemas híbridos são aqueles cujas especificações de controle podem ser descritas por linguagens formais aceitas por autômatos finitos. Exemplos incluem problemas de segurança, onde o controlador garante que a planta não entrará em uma região de operação não segura (estado proibido), ou garantindo por exemplo que um único recurso não seja acessado simultaneamente por dois métodos (exclusão mútua), como no caso em que dois robôs têm acesso a uma mesma região e que se deseja evitar a colisão dos mesmos. Outra especificação comum nesta abordagem é a de alcançabilidade, na qual o controlador deve levar a planta de uma região (ou estado) inicial de operação para uma outra desejada; este é o caso por exemplo de um procedimento de *start-up* em uma planta química.

É importante ressaltar que a abordagem de controle supervisório não tem o intuito de tratar problemas de controle contínuo, tais como problemas de estabilidade. Assim, qualquer ação de controle contínuo deve ser considerada como parte da planta a ser controlada. Desta forma, a planta pode ser constituída por uma ou mais malhas de controle (processo + controlador) contínuo.

A estrutura apresentada na Figura 2.2 foi proposta por James A. Stiver, Panos J. Antsaklis e Michael D. Lemmon em (Antsaklis et al., 1993) e depois foi considerada em vários trabalhos publicados por estes autores (Stiver et al., 1995a; Stiver et al., 1995b; Stiver et al., 1996a; Stiver et al., 1996b; Koutsoukos et al., 2000). Este modelo é essencialmente o mesmo que o proposto por Anil Nerode e Wolf Kohn em (Nerode e Kohn, 1993).

Abordagens similares baseadas em aproximações da planta por um SED foram utilizadas por Caines e Wei (1998), Cury et al. (1998), Raisch e O'Young (1998) e Moor, Raisch e

Davoren (2001). A principal diferença da metodologia de modelagem empregada por Cury et al. (1998) em relação àquelas utilizadas nos demais trabalhos supracitados é que ela permite que a modelagem do sistema híbrido seja feita por intermédio de diagramas de blocos. Tal metodologia utiliza-se do formalismo de sistemas condição/evento (Sreenivas e Krogh, 1991) e foi proposta por Krogh (1993). Ela é baseada na definição formal de blocos individuais de tempo contínuo, os quais possuem entradas e saídas que podem ser interligadas de forma a criar modelos para sistemas híbridos através de diagramas de blocos. Tendo em vista que esta será a abordagem de modelagem utilizada ao longo deste trabalho, no Capítulo 3 apresentar-se-á a mesma de forma detalhada. Muitos outros trabalhos utilizaram o paradigma de sistemas condição/evento para a modelagem de sistemas híbridos (Niinomi e Krogh, 1995; Chutinan, 1999; Chutinan e Krogh, 1999a; Chutinan e Krogh, 1999b; Cury e Krogh, 1999; Silva et al., 2000; Chutinan e Krogh, 2001; González et al., 2001; Silva et al., 2001).

Os pesquisadores Raisch, Moor, O'Young, Davoren e Itigin também abordam o tema controle supervisório de sistemas híbridos (Moor e Raisch, 1999; Raisch et al., 2000; Moor, Davoren e Raisch, 2001; Moor, Raisch e Davoren, 2001; Moor, Davoren e Raisch, 2002; Moor e Raisch, 2002; Moor, Raisch e O'Young, 2002). Em seus trabalhos, estes autores tratam o problema usando a chamada *"teoria comportamental de sistemas"*, desenvolvida por Willems (1991). Nesta abordagem, o comportamento de um modelo é dado pelo conjunto de todas as trajetórias de seus sinais externos ao longo do tempo (Raisch e O'Young, 1998). Assim, após capturado o comportamento da planta, a realimentação dinâmica para esta planta resulta essencialmente na intersecção dos conjuntos que representam os comportamentos da planta e do controlador. Desta forma, para que o problema de controle supervisório tenha solução é necessário que esta intersecção, além de ser não vazia, esteja contida no conjunto que representa o *comportamento especificado* (desejado) para o sistema sob supervisão.

Existem ainda muitas outras metodologias de modelagem, sendo que algumas delas (Branicky et al., 1994; Lygeros, 1996; Branicky et al., 1998) buscam capturar tanto dinâmicas contínuas quanto dinâmicas discretas. Em sua tese de doutorado, Branicky (1995) introduziu um modelo geral para sistemas dinâmicos híbridos. Este modelo consiste em coleções de sistemas dinâmicos interagindo, cada um evoluindo no espaço de estados contínuo, com chaveamento entre estes sistemas ocorrendo quando uma variável de estado intercepta subconjuntos específicos de seu espaço de estados. Sendo assim, este modelo possibilita expressar saltos nos estados contínuos.

Outras abordagens de modelagem de sistemas híbridos foram apresentadas por Pettersson e Lennartson em (Pettersson e Lennartson, 1995), (Lennartson et al., 1996), (Pettersson, 1999) e (Johansson et al., 2004).

2.3.2 Perspectiva da Ciência da Computação

Após a expansão de técnicas de controle supervisório de SEDs, no final dos anos 80 (Wonham e Ramadge, 1987; Wonham e Ramadge, 1988), estudiosos da computação buscaram estender modelos puramente discretos de forma a incluir dinâmicas contínuas. Como resultado inicial desta extensão foram propostos os autômatos diferenciais (Tavernini, 1987) e os diagramas de estados híbridos (ou *hybrid statecharts*) (Maler et al., 1992).

A seguir foram propostos os autômatos temporizados (Alur e Dill, 1990; Alur e Dill, 1994), em cujos estados discretos foram inseridas dinâmicas do tipo relógio (na forma $\dot{x} = 1$). Assim, o comportamento contínuo resultante é monotonicamente (monotonamente) crescente com uma taxa constante e o mesmo pode ser reinicializado nos instantes de ocorrência de eventos discretos. Em um passo posterior, foram introduzidos os autômatos híbridos (Alur et al., 1993), os quais não possuem a restrição de monotonicidade e ainda permitem relógios com taxas distintas, podendo ser considerados então como uma generalização dos anteriores. Vale destacar que existem algumas classes de autômatos híbridos, tais como os autômatos de múltiplas taxas (*multirate automata*), que capturam dinâmicas na forma $\dot{x} = c$, onde c é uma constante; os chamados autômatos retangulares (Puri e Varaiya, 1994; Henzinger et al., 1995) nos quais as dinâmicas contínuas são modeladas por inclusões diferenciais da forma $\dot{x} \in [a, b]$, onde a e b são constantes; os autômatos híbridos lineares (Henzinger, 1996), que capturam dinâmicas na forma $A\dot{x} < b$; e ainda os denominados autômatos híbridos entrada/saída (*hybrid I/O automata*) (Lynch et al., 1996; Lynch et al., 2003). Para maiores detalhes ver (Alur et al., 1995) e (Lygeros et al., 2003).

Existem ainda outros paradigmas para a modelagem de sistemas híbridos que podem ser considerados extensões dos autômatos temporizados, tais como aqueles introduzidos por Nicollin et al. (1993), os quais são muito semelhantes ao formalismo dos autômatos híbridos, e por Alur et al. (1995).

Além destes, muitos formalismos matemáticos foram propostos para a modelagem de sistemas híbridos. Eles incluem redes de Petri temporizadas (He e Lemmon, 1998; Koutsoukos e Antsaklis, 1999), redes de Petri diferenciais (Demongodin e Koussoulas, 1998), redes de Petri

híbridas (Le Bail et al., 1991; Pettersson e Lennartson, 1995; David, 1997; David e Alla, 2001), a representação por transição de estado (state-transition network representation) (Avraam et al., 1998), a representação por intermédio de *bond graphs* (Mosterman, 1997), entre outros.

A maior parte da pesquisa desenvolvida junto à comunidade de ciência da computação tem sido direcionada para o contexto de verificação formal (Chutinan, 1999; Clarke et al., 2003; Chutinan e Krogh, 2003; Tomlin et al., 2003) utilizando tanto a técnica chamada de verificação de modelos (*model checking*) (Alur, Henzinger e Ho, 1996; Henzinger, Ho e Wong-Toi, 1998; Henzinger e Majumdar, 2000; Fehnker, 2002; Amla et al., 2003; Ábrahám et al., 2005), a qual testa exaustivamente todas as trajetórias do sistema, quanto a técnica dedutiva de prova de teoremas (Bjorner et al., 1995; Manna e Sipma, 1998), que prova a especificação através da indução sobre todas as trajetórias. Dentro da área de verificação de sistemas híbridos um tema de pesquisa muito importante tem sido a verificação de propriedades de segurança em sistemas híbridos. Deve-se responder de forma segura perguntas do tipo: *um determinado estado ou uma configuração insegura é alcançável a partir de uma dada configuração inicial?* Para uma revisão sobre o tema sugere-se a leitura de (Chutinan e Krogh, 2003) e (Tomlin et al., 2003).

Entre os avanços mais significativos na perspectiva da ciência da computação estão os resultados extensivos sobre decidibilidade em problemas de verificação para várias classes de sistemas híbridos (Puri e Varaiya, 1994; Henzinger, Kopke, Puri e Varaiya, 1998; Lafferriere et al., 1999; Miller, 2000). Sabe-se hoje em dia que a introdução de dinâmicas contínuas, mesmo que muito simples, em um modelo de autômato híbrido pode fazer com que problemas de verificação se tornem indecidíveis. Segundo Krogh (2000), estes resultados deixaram claro que não se deve esperar muito das ferramentas de computador para a análise de sistemas híbridos.

2.4 Ferramentas Computacionais para Sistemas Híbridos

Atualmente, existem várias ferramentas computacionais (algumas ainda em fase de desenvolvimento), tanto comerciais quanto acadêmicas, que podem ser utilizadas para modelagem, simulação, projeto e verificação de sistemas híbridos. Nesta seção, citam-se algumas das principais ferramentas e apresenta-se uma breve descrição das mesmas. Não se faz, entretanto, uma descrição detalhada e nem tampouco uma análise comparativa entre elas. Para uma descrição detalhada destas e de outras ferramentas, bem como um estudo comparativo das

mesmas, consultar os trabalhos de Kowalewski et al. (1999), Mosterman (1999), van Beek e Rooda (2000), Silva et al. (2001), Carloni et al. (2004) e Pinto et al. (2005).

2.4.1 Ferramentas para Modelagem, Simulação e Projeto

GRAIL: consiste em uma biblioteca de funções em C++ para manejo de autômatos, expressões regulares e linguagens (Raymond e Wood, 1995; Raymond e Wood, 1996). Ela foi ampliada por González (2000) para incluir funções de controle supervisório e também por da Cunha (2003) para incluir funções de controle supervisório hierárquico. Ela foi estendida por Rodrigues (2004) para tratar problemas de controle supervisório modular para sistemas híbridos, tema da presente tese. Nesta ferramenta, utiliza-se o paradigma de sistemas condição/evento (Sreenivas e Krogh, 1991; Krogh, 1993) como base para a modelagem de sistemas híbridos. A ferramenta, com suas extensões, pode ser encontrada em `http://www.das.ufsc.br/~cury/ensino/`.

HYVISUAL (Hybrid System Visual Modeler): desenvolvida no EECS (Electrical Engineering and Computer Sciences Department) da Universidade da Califórnia - Berkeley, é uma ferramenta baseada em diagramas de blocos constituída por um editor e simulador para sistemas contínuos e sistemas híbridos (Brooks et al., 2004; Lee e Zheng, 2005). HYVISUAL é uma ferramenta desenvolvida sobre uma plataforma chamada PTOLEMY II (Davis et al., 1999; Eker et al., 2003), e pode ser gratuitamente obtida em `http://ptolemy.eecs.berkeley.edu/hyvisual/`.

MODELICA: desenvolvida pela *ModelicaHP*, é uma linguagem orientada a objetos que serve para a modelagem de sistemas físicos que possuem fenômenos híbridos. Ela suporta vários formalismos, tais como equações diferenciais ordinárias (EDO), equações diferenciais-algébricas (differential-algebraic equations - DAE), *bond graphs*, autômatos de estados finitos, redes de Petri, etc. (Mattsson et al., 1999; Tiller, 2001; Fritzson, 2004). Pacotes e ferramentas comerciais compatíveis, tais como DYMOLA (Dynamic Modeling Laboratory) da *Dynasim* e MATHMODELICA da *MathCore* podem ser usados em conjunto com o MODELICA para realizar a simulação de sistemas híbridos. `http://www.modelica.org/`.

SCICOS (SCIlab Connected Object Simulator): desenvolvida no INRIA, na França, é uma ferramenta para modelagem e simulação de sistemas dinâmicos que possuem tanto dinâmicas contínuas quanto dinâmicas discretas e dirigidas a eventos. Trata-se de uma ferramenta

gratuita que compõe o pacote SCILAB. Ela inclui um editor gráfico para a construção de modelos através da interconexão de blocos (Nikoukhah e Steer, 1997) e pode ser obtida em `http://www.scicos.org`.

SIMULINK/STATEFLOW: desenvolvido e comercializado pela *Mathworks*, consiste em um pacote do MATLAB para a modelagem e simulação visual de sistemas que possuem dinâmicas de tempo contínuo, de tempo discreto e dinâmicas dirigidas por eventos. Modelos complexos são facilmente construídos por intermédio da interconexão de blocos já existentes numa biblioteca ou de blocos criados pelo usuário. `http://www.mathworks.com/`.

2.4.2 Ferramentas para Verificação Formal

CHARON: sigla para *coordinated control, hierarchical design, analysis and run-time monitoring of hybrid systems*, trata-se de uma linguagem para a especificação (modelagem) modular e hierárquica de sistemas híbridos (Alur et al., 2000; Alur et al., 2001; Alur et al., 2003). Esta linguagem é desenvolvida no CIS-UPENN (Computer and Information Science Department - University of Pennsylvania). `http://www.cis.upenn.edu/mobies/charon/`.

CHECKMATE: desenvolvida no ECE (Electrical and Computer Engineering Department) da Universidade de Carnegie Mellon (CMU), nos Estados Unidos, esta é uma ferramenta baseada no MATLAB que serve para a simulação e verificação de sistemas híbridos. Os modelos são construídos usando a interface gráfica SIMULINK e parâmetros e as especificações são inseridas diretamente no SIMULINK ou por intermédio de arquivos definidos pelo usuário (*m-files*). Esta ferramenta possibilita a verificação de sistemas híbridos através da obtenção de aproximações para seu modelo discreto. Detalhes sobre a ferramenta são dados em (Chutinan, 1999; Silva et al., 2000) e outras informações podem ser obtidas em `http://www.ece.cmu.edu/~webk/checkmate`.

SHIFT: é uma linguagem de programação para a descrição de redes dinâmicas de autômatos híbridos que vem sendo desenvolvida no projeto PATH (Partners for Advanced Transit and Highways) na Universidade da Califórnia - Berkeley (Deshpande et al., 1997; Deshpande et al., 1998). Esta linguagem é usada principalmente para a descrição e simulação de sistemas híbridos complexos cuja configuração varia com o tempo. Muito embora a motivação inicial tenha sido a especificação e a análise de aplicações automotivas – rodovias e auto-

móveis inteligentes (Göllü e Kourjanski, 1997), esta linguagem tem sido utilizada em vários domínios de aplicação, tal como em Sistemas de Gerenciamento de Tráfego Aéreo e em sistemas de gerenciamento de redes. O nome SHIFT é resultado da permutação de letras de HSTIF (Hybrid Systems Tool Interchange Format). Outras informações podem ser obtidas em `http://path.berkeley.edu/shift/`.

HYSDEL (HYbrid system DEscription Language): distribuída pelo laboratório de Controle Automático do Instituto Federal Tecnológico da Suíça, é uma linguagem para descrição de sistemas híbridos. Entretanto, esta linguagem limita-se à descrição de sistemas híbridos lineares e de tempo discreto (Potoènik et al., 2003; Bemporad e Morari, 1999). `http://control.ee.ethz.ch/~hybrid/hysdel/`.

PHAVER (Polyhedral Hybrid Automaton VERifyer): ferramenta para verificação de propriedades em sistemas híbridos modelados por intermédio de autômatos híbridos poliedrais (Frehse, 2005b). Esta ferramenta está sendo desenvolvida por Goran Frehse em seu doutoramento (Frehse, 2005a), o qual está sendo realizado na Universidade de Carnegie Mellon - USA, sob a orientação do prof. Bruce Krogh. `http://www.cs.ru.nl/~goranf/`.

Outras ferramentas para verificação de sistemas híbridos são MASACCIO (Henzinger, 2000), UPPAAL (Behrmann et al., 2004), d/dt (Asarin et al., 2002), e HYTECH (Henzinger et al., 1997).

Existe ainda uma ferramenta denominada HYBRID TOOLBOX que está sendo desenvolvida no Depto. de Engenharia da Informação da Universidade de Siena, na Itália, sob a coordenação de Alberto Bemporad, e que se propõe tanto à modelagem e simulação quanto à verificação de sistemas híbridos. Esta ferramenta também é voltada para o MATLAB/SIMULINK. Informações sobre esta ferramenta podem ser obtidas em `http://www.dii.unisi.it/hybrid/toolbox`.

Dentre as ferramentas acima citadas, utilizam-se o GRAIL e o CHECKMATE neste trabalho. Conforme será estudado no Capítulo 3, a obtenção de um modelo de estados finitos que represente o comportamento lógico exato da planta híbrida normalmente é uma atividade extremamente difícil e então utilizam-se aproximações conservadoras para estes modelos (Cury et al., 1998). A principal utilização do CHECKMATE neste trabalho consiste na geração de tais aproximações, de forma que elas possam ser utilizadas no GRAIL para realizar o projeto do supervisor.

2.5 Exemplos de Aplicação de Sistemas Híbridos

Os sistemas híbridos proporcionam um rico contexto para a definição de vários tipos de problemas de controle. Sendo assim, muitos autores vislumbraram a utilização do paradigma de sistemas híbridos para a resolução de problemas específicos. Enumeram-se, a seguir, algumas das aplicações nas quais são utilizadas abordagens de sistemas híbridos.

- aplicações automotivas, tais como em sistemas de rodovias e automóveis inteligentes (Varaiya, 1993; Godbole e Lygeros, 1994; Lygeros et al., 1998; Balluchi et al., 2000; Horowitz e Varaiya, 2000; Girard et al., 2001; Altafini et al., 2002; Stursberg et al., 2004) e na modelagem e controle de tráfego urbano (Cury e Garcia, 2004; Garcia e Cury, 2004);
- sistemas de segurança crítica (Livadas, 1997; Livadas e Lynch, 1998), tais como processos nucleares, sistemas de controle de tráfego aéreo (Tomlin et al., 1998; Tomlin, 1998; Bicchi e Pallottino, 2000) e sistemas de controle de aeronaves (Livadas et al., 2000), tais como o *air traffic alert and collision avoidance system*;
- sistemas de manufatura (Brandin, 1996; Pepyne e Cassandras, 2000);
- robótica (Song et al., 2000; Branicky e Chhatpar, 2002; Egerstedt, 2000; Egerstedt e Hu, 2002);
- eletrônica de potência (Miranda e Lima, 2003; Senesky et al., 2003; Lima e Miranda, 2004);
- redes de distribuição de energia (Kloas et al., 1995; Ferrari-Trecate et al., 2002); e
- controle de processos químicos (Tittus e Lennarston, 1998; Fritz et al., 1999; Thevenon, 2000; Engell et al., 2000; Raisch et al., 2000; Potocnik et al., 2002; van Beek et al., 2002).

2.6 Conclusões

De uma forma geral, os sistemas híbridos são aqueles que possuem dinâmicas contínuas e dinâmicas discretas (ou a eventos discretos), sendo que, além de coexistirem, estas dinâmicas também podem interagir. Pela própria natureza (híbrida) destes sistemas, o estudo dos mesmos requer conhecimentos das áreas de sistemas contínuos e também de sistemas (a eventos)

discretos. Entretanto, apesar de complexos, os sistemas híbridos se mostram bastante adequados para representar uma grande quantidade de aplicações nas mais diversas áreas. Assim, o estudo de problemas relacionados aos sistemas híbridos têm despertado grande interesse junto às comunidades de controle e de ciência da computação, o que levou ao surgimento de diversos enfoques e abordagens bastante distintos para tratar cada um destes problemas.

Todos estes fatores contribuem para que os resultados obtidos a respeito do tema ainda estejam pouco sedimentados e, conseqüentemente, para que ainda existam poucos resultados clássicos. Um resultado importante na teoria de sistemas híbridos mostra que mesmo sistemas muito simples podem exibir comportamentos complexos (Ramadge, 1990; Chase et al., 1993) e que problemas de verificação podem ser indecidíveis (Henzinger, Kopke, Puri e Varaiya, 1998). Sendo assim, a síntese de controladores para sistemas híbridos genéricos provavelmente sempre necessitará da introdução de aproximações e heurísticas para evitar que se recaia em cálculos intratáveis.

Neste capítulo, introduzem-se os sistemas híbridos e faz-se uma breve revisão histórica sobre os mesmos, mostrando a evolução de seu estudo ao longo das últimas décadas. Apresentam-se ainda as principais abordagens existentes para o tratamento de problemas relacionados aos sistemas híbridos sob as perspectivas das comunidades de controle e de ciência da computação. Por fim, citam-se alguns exemplos de aplicação de sistemas híbridos e enumeram-se uma série de ferramentas computacionais que servem para modelagem, análise, simulação, verificação e/ou síntese deste tipo de sistema.

Capítulo 3

Controle Supervisório de Sistemas Híbridos

Neste capítulo apresenta-se a formulação e solução de um problema de controle supervisório para uma classe de sistemas híbridos. Uma das principais contribuições do mesmo consiste em estender os resultados apresentados em Cury et al. (1998), González (2000) e González et al. (2001) de forma a permitir a resolução do problema tratado aqui. Destaca-se a extensão feita para permitir a abordagem por linguagens marcadas, abordagem esta não considerada nos trabalhos supracitados.

3.1 Introdução

Os sistemas híbridos, de uma forma geral, são sistemas que combinam dinâmicas contínuas e dinâmicas discretas. Na classe de sistemas híbridos considerada neste trabalho, a dinâmica discreta é definida por eventos gerados quando as variáveis do espaço de estados contínuo (ou funções destas) alcançam uma superfície de limiar, forçando então transições no estado discreto. A dinâmica contínua, por sua vez, é determinada por uma condição discreta, função do estado discreto atual do sistema. O objetivo consiste em sintetizar um supervisor que restrinja as transições de estado discreto do sistema híbrido de tal forma que todas as possíveis seqüências de eventos estejam contidas em um dado conjunto de seqüências de eventos especificado pelo projetista (linguagem desejada para o sistema sob supervisão). Além disso, o supervisor deve ser não bloqueante, isto é, deve permitir que a planta sempre possa completar alguma tarefa. Assim, neste trabalho o controlador é um sistema de estado discreto que é utilizado

para supervisionar o comportamento de uma planta híbrida. Segundo Krogh (2000), esta é a situação geralmente encontrada em sistemas de controle industriais em que são utilizados Controladores de Lógica Programável (CLPs) para a realização da tarefa de controle. Neste capítulo, o problema de controle supervisório de sistemas híbridos é apresentado por meio do formalismo de sistemas condição/evento (C/E). Introduzidos por Sreenivas e Krogh (1991), os sistemas C/E fornecem uma forma de se definir sistemas em tempo contínuo pela interconexão de subsistemas com sinais de entrada e saída discretos. Sendo assim, os sistemas híbridos são representados por intermédio de diagramas de blocos e de fluxos de sinais discretos. Vale ressaltar que embora esteja sendo utilizado aqui para a modelagem de sistemas híbridos, o formalismo de sistemas C/E pode ser utilizado também para a modelagem de sistemas a eventos discretos (Garcia e Cury, 2002; Leal e Cury, 2002). Para estes sistemas, muitas vezes a abordagem por sistemas C/E mostra-se mais natural e intuitiva que aquelas oferecidas por formalismos estritamente discretos, tais como autômatos e linguagens formais. O problema de controle supervisório tratado neste trabalho consiste em uma generalização dos problemas considerados por Cury et al. (1998) e Koutsoukos et al. (2000). Problemas de controle supervisório semelhantes são considerados também por Moor et al. (1998) e Raisch e O'Young (1998) para sistemas com dinâmica contínua em tempo discreto.

O restante do capítulo está organizado como segue. Na próxima seção, o problema de controle supervisório de sistemas híbridos de tempo contínuo é formulado utilizando-se o formalismo de sistemas condição/evento (Sreenivas e Krogh, 1991). Na Seção 3.3 mostra-se como o problema anteriormente formulado pode ser convertido em um problema clássico de síntese de supervisores para sistemas a eventos discretos (SEDs). Define-se então um modelo de SEDs para representar os comportamentos seqüenciais de entrada-saída admissíveis para a planta híbrida e mostra-se que o supervisor obtido para o problema equivalente consiste em uma solução para o problema original. Ao longo de todo capítulo são apresentados exemplos de forma a facilitar o entendimento dos conceitos introduzidos. A Seção 3.4 encerra o presente capítulo.

3.2 Formulação do Problema

Considere o esquema de controle supervisório mostrado na Figura 3.1. A planta híbrida $\mathcal{H}$ é um sistema C/E (Sreenivas e Krogh, 1991) dado pela interconexão de um subsistema

de dinâmica contínua, $\mathcal{H}_c$, e outro de dinâmica discreta, $\mathcal{H}_d$. Sem perda de generalidade, assume-se que existe apenas um subsistema de cada tipo. Múltiplos subsistemas do mesmo tipo podem ser reduzidos a um único subsistema através da utilização de vetores de sinais de entrada e saída. A seguir descreve-se o comportamento de cada uma das partes da planta híbrida.

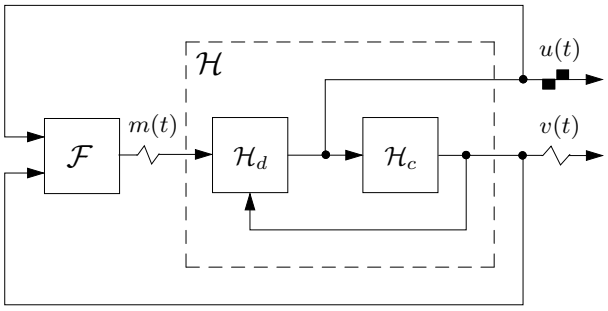


Figura 3.1: Esquema de controle supervisório para sistemas híbridos.

Subsistema Contínuo

Conforme mostrado na Figura 3.2, o subsistema contínuo é formado por dois tipos de subsistemas: um *sistema contínuo chaveado* e um *gerador de eventos de limiar*. O sinal de entrada para o subsistema contínuo é um sinal condição, $u(\cdot)$, um sinal constante por partes, contínuo à direita e com limites à esquerda (ver Figura 3.3(a)), assumindo valores sobre um conjunto finito de condições U. O espaço de todos os sinais condição $u(\cdot)$ em $[0, \infty)$ é denotado por $\mathcal{U}$. As dinâmicas contínuas da planta híbrida são definidas por uma trajetória contínua

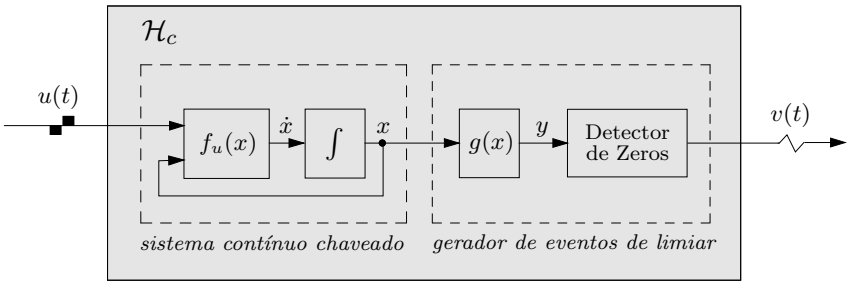


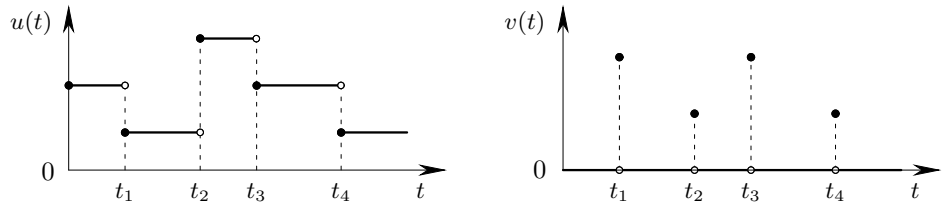
Figura 3.2: Subsistema contínuo $\mathcal{H}_c$.

de estados, $x(\cdot)$, que evolui em $X = \mathbb{R}^n$ e que satisfaz uma equação de estados da forma $\dot{x}(t) = f_{u(t)}(x(t))$, onde $f_{u(t)} : \mathbb{R}^n \to \mathbb{R}^n$ para todo $u(t) \in U$. Assim, a seleção da dinâmica contínua (representada pela equação de estados) é feita a cada instante pela condição $u(t)$. O valor inicial da trajetória de estados, $x(0)$, pertence ao conjunto de estados iniciais $X_0 \subseteq \mathbb{R}^n$.

O conjunto de todas as possíveis trajetórias de estado para um dado sinal de entrada $u(\cdot) \in \mathcal{U}$, iniciando em qualquer estado do conjunto $X' \subset X_0$, é denotado por $\mathcal{X}_{u(\cdot)}(X')$. A função $g : X \to \mathbb{R}^k$ é aplicada ao sinal $x(\cdot)$ para produzir o sinal de saída $y(\cdot)$ e é tal que cada componente g_i define uma hipersuperfície [1] de limiar $T_i = \{x : g_i(x) = 0\}$. O sinal resultante $y(\cdot) = g(x(\cdot))$ é aplicado a um *detector de zeros* de forma a gerar o sinal de saída evento, o qual é definido para $t > 0$ como $v(t) = [v_1(t) \quad v_2(t) \quad \dots \quad v_k(t)]^T$, onde cada componente $v_i(t)$ $(i = 1, \dots, k)$ é dado por:

$$v_i(t) = \begin{cases} 1 & \text{se} \quad y_i(t) = 0 \bigwedge (\exists \Delta > 0)(\forall \delta \in (0, \Delta)) : y_i(t - \delta) < 0 \\ 0 & \text{de outra forma} \end{cases} \tag{3.1}$$

Naqueles instantes em que qualquer $v_i(t) = 1$ (quando uma superfície de limiar é cruzada), diz-se que ocorreu um *evento de limiar*. Para $t = 0$ considera-se a existência de um *evento de inicialização* $v(0) = v_0$, um evento que é associado à escolha não determinística do estado inicial $x(0)$. Assim, o sinal evento $v(\cdot)$ assume valores não nulos sobre o conjunto $V^+ = \{v_0\} \cup V$ em pontos isolados do tempo, onde $V = \{0, 1\}^k - \{0\}^k$ é o *conjunto de eventos de limiar*, um conjunto de vetores não nulos de dimensão m cujos elementos assumem apenas os valores 0 e 1. O espaço de todos os sinais evento $v(\cdot)$ em $[0, \infty)$ é denotado por $\mathcal{V}$. A Figura 3.3(b) ilustra a representação de um sinal evento, sendo que diferentes amplitudes no sinal evento foram usadas para simbolizar distintos eventos de limiar. Assim, pode-se imaginar, por exemplo, que $v(t_1) = [1 \quad 0]^T$ e $v(t_2) = [0 \quad 1]^T$.



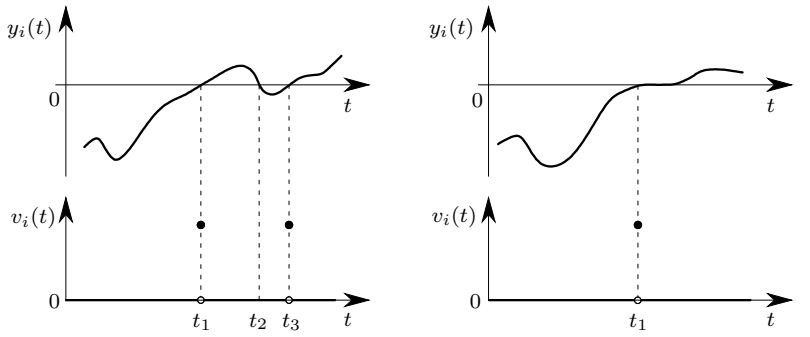
(a) Sinal condição (b) Sinal evento

Figura 3.3: Representação dos sinais condição e evento.

Observe que na equação (3.1) a condição dada por $y_i(t - \delta) < 0$ faz com que um evento de limiar seja gerado somente quando uma superfície de limiar T_i é alcançada por baixo, ou seja,

[1] O termo hipersuperfície é usado para designar, de forma genérica, uma superfície de dimensão $n - 1$ colocada em um espaço de dimensão n. Uma hipersuperfície é, portanto, o conjunto de soluções para a equação $f(x_1, \dots, x_n) = 0$.

o detector de zeros (e conseqüentemente a sinalização de eventos) é unidirecional. A Figura 3.4(a) ilustra esse aspecto; note que nos instantes t_1 e t_3 é gerado um evento de limiar, mas que no instante t_2 nenhum evento é gerado. É importante destacar que poder-se-ía utilizar a sinalização bidirecional, mas que escolheu-se esta forma por ser a mais comumente encontrada em aplicações práticas. Note ainda que a inequação $y_i(t-\delta) < 0$ deve ser satisfeita para um $\delta > 0$, ou seja, para que um evento de limiar seja gerado no instante t é necessário que $y_i < 0$ durante algum tempo antes de $y_i(t)$ se tornar nulo. Essa restrição impede que inúmeras descontinuidades no sinal evento $v(\cdot)$ sejam geradas quando $y_i(t)$ permanece nulo por um certo tempo (ver Figura 3.4(b)).



(a) Aspecto unidirecional (b) Deslizamento sobre a superfície

Figura 3.4: Aspectos na geração de eventos de limiar.

Note ainda que um evento de limiar distinto é gerado quando mais de uma superfície de limiar é alcançada em um dado instante t. Neste caso, a informação sobre a localização do estado contínuo do sistema é conhecida com maior precisão que quando qualquer uma das superfícies é alcançada individualmente. Sendo assim, não se considera a ocorrência simultânea de eventos de limiar.

Assume-se que a condição de entrada para $\mathcal{H}_c$ pode ser alterada (atualizada) somente quando um evento de limiar é observado. O subsistema contínuo $\mathcal{H}_c$ é definido como um subconjunto do *produto cartesiano síncrono* de $\mathcal{V}$ e $\mathcal{U}$, denotado por $\mathcal{V} \otimes \mathcal{U}$, o qual representa o conjunto de todos os pares $(v(\cdot), u(\cdot)) \in \mathcal{V} \times \mathcal{U}$ tais que descontinuidades em $u(\cdot)$ ocorrem apenas nos instantes em que $v(\cdot)$ é não nulo. Um par $(v(\cdot), u(\cdot)) \in \mathcal{H}_c$ se e somente se existe uma trajetória de estados $x(\cdot) \in \mathcal{X}_{u(\cdot)}(X_0)$ tal que o sinal evento de saída resultante é $v(\cdot)$.

Subsistema Discreto

O subsistema $\mathcal{H}_d$ é um sistema de dinâmica puramente discreta que restringe a gama de

sinais de entrada que podem ser aplicadas a $\mathcal{H}_c$, modelando restrições físicas do subsistema contínuo (restrições nas possibilidades de chaveamento de dinâmicas). Ele é dirigido por dois sinais evento, $v(\cdot)$ e $m(\cdot)$, onde $m(\cdot)$ é um sinal de controle externo que assume valores sobre o conjunto $M = 2^U$. O conjunto de todos os sinais evento $m(\cdot) \in [0, \infty)$ é denotado por $\mathcal{M}$. Segundo Sreenivas e Krogh (1991), mudanças na saída condição de um sistema C/E só podem ocorrer nos instantes de descontinuidade no sinal entrada evento deste mesmo sistema. Como $\mathcal{H}_d$ possui duas entradas evento, estas são síncronas e assim $\mathcal{H}_d$ é definido como um subconjunto do produto cartesiano síncrono de $\mathcal{V}$, $\mathcal{M}$ e $\mathcal{U}$, denotado por $\mathcal{V} \otimes \mathcal{M} \otimes \mathcal{U}$, o qual representa o conjunto de todos os sinais $(v(\cdot), m(\cdot), u(\cdot)) \in \mathcal{V} \times \mathcal{M} \times \mathcal{U}$ tais que descontinuidades em $u(\cdot)$ ocorrem apenas nos instantes em que $v(\cdot)$ é não nulo e $m(\cdot)$ é não vazio. Note que o evento de inicialização é usado para determinar a gama de sinais que podem ser aplicados a $\mathcal{H}_c$ no instante $t = 0$, quando nenhum evento de limiar ocorreu.

A planta híbrida $\mathcal{H} \subseteq \mathcal{V} \otimes \mathcal{M} \otimes \mathcal{U}$ é obtida pela conexão cascata e realimentação de $\mathcal{H}_c$ e $\mathcal{H}_d$ (ver Figura 3.1), seguindo (Sreenivas e Krogh, 1991).

O supervisor $\mathcal{F}$ observa os sinais condição $u(\cdot)$ e evento $v(\cdot)$ da planta híbrida $\mathcal{H}$ e, em resposta, aplica uma entrada de controle $m(\cdot)$ ao subsistema discreto $\mathcal{H}_d$ de forma a restringir a gama de possíveis condições de entrada aplicadas ao subsistema contínuo $\mathcal{H}_c$. Vale enfatizar que se o sinal de entrada $m(\cdot)$ para a planta híbrida $\mathcal{H}$ puder chavear livremente em todo e qualquer instante, então a análise do comportamento do sistema híbrido poderá se tornar intratável devido a mudanças constantes na dinâmica da planta. Assim, o cenário de interesse é aquele em que descontinuidades no sinal evento $m(\cdot)$ (i.e. $m(t) \neq \emptyset$) ocorrem somente nos instantes de descontinuidade no sinal evento $v(\cdot)$ (i.e. $v(t) \neq 0$), o que faz com que as descontinuidades em $m(\cdot)$ e em $v(\cdot)$, e conseqüentemente em $u(\cdot)$, sejam síncronas. Desta forma, cada valor não vazio assumido por $m(t)$ deve ser interpretado como uma atualização (nos instantes t) no conjunto de condições que $\mathcal{H}_d$ pode aplicar a $\mathcal{H}_c$.

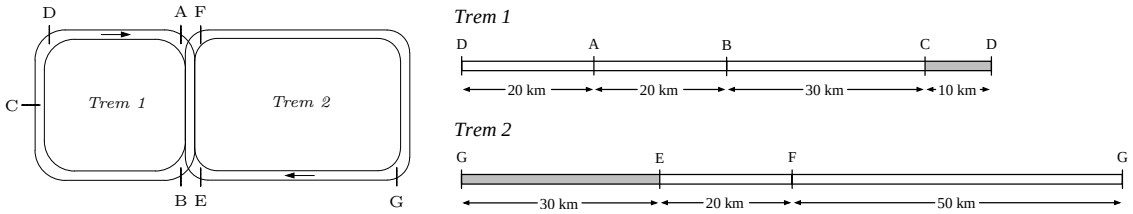
Por fim, considera-se a hipótese de sistema *não Zeno*, ou seja, que a realimentação da Figura 3.1 não leva ao *chattering*, isto é, em qualquer intervalo finito de tempo o número de eventos de limiar é sempre finito.

Para modelar o comportamento livre da planta híbrida sem a ação do supervisor, considera-se que $\forall (v(\cdot), m(\cdot), u(\cdot)) \in \mathcal{H},\ m(t) = U$ para todo $t \in [0, \infty)$ tal que $v(t) \neq 0$, isto é, considera-se que restrições sobre a gama de condições que $\mathcal{H}_d$ pode aplicar a $\mathcal{H}_c$ são impostas apenas por $\mathcal{H}_d$.

A seguir são apresentados dois exemplos de modelagem de sistemas híbridos em malha aberta. No primeiro exemplo a planta possui restrições físicas impostas à escolha da dinâmica contínua. Assim, para alguns eventos na sua entrada, o subsistema discreto pode ter como saída apenas algumas condições. Neste caso, $\mathcal{H}_d$ permite a introdução de informações sobre restrições nas possibilidades de chaveamento. No segundo exemplo a planta não possui restrições físicas (restrições nas possibilidades de chaveamento). Assim, para cada evento em sua entrada, o subsistema $\mathcal{H}_d$ tem como saída todos as possíveis condições, sem restrições. Neste caso, o subsistema $\mathcal{H}_d$ é utilizado simplesmente como artifício para possibilitar a modelagem do comportamento da planta livre, incluindo no modelo da planta todas as possíveis ações de controle. Vale salientar que este último trata-se de um caso particular do primeiro.

Exemplo 3.1 ***: Sistema de Trens***

Considere o sistema de dois trens sobre trilhos cíclicos que compartilham um trecho, conforme ilustrado na Figura 3.5. Este exemplo é uma ilustração de um problema de controle mais amplo que é o controle de malhas ferroviárias e foi proposto por González (2000) e utilizado também em (González et al., 2001).



(a) Sistema de Trens (b) Trilhos

Figura 3.5: Exemplo de sistema de trens.

No sistema em questão, existem sensores sobre os trilhos que detectam a passagem dos trens pelos pontos A, B, C e D (trem 1) e E, F e G (trem 2). Os trens podem ser comandados a trafegar em dois modos distintos de velocidade: rápido *e* devagar*, os quais serão denotados por* $\boldsymbol{r}$ *e* $\boldsymbol{d}$*, respectivamente. Entretanto, somente nos instantes de sinalização da passagem dos trens por estes pontos é que pode haver um comando de mudança de velocidade. Uma característica importante deste sistema é que o modo "devagar" só pode ser comandado no ponto C (trecho $\overline{CD}$) para o trem 1, e no ponto G (trecho $\overline{GE}$) para o trem 2 (vide trechos sombreados na Figura 3.5(b)). Esta limitação implica restrição nas possibilidades de chaveamento, o que caracteriza este exemplo como ilustrativo para a classe de sistemas híbridos em questão.*

Na modelagem deste sistema, a trajetória de estados em tempo contínuo será considerada como sendo as posições dos trens medidas sobre os trilhos. A medida da posição dos trens é tal que a variável associada é contínua e limitada. Escolhem-se os pontos D e G como sendo as origens das trajetórias, e os pontos B e F como pontos médios das mesmas, para os trens 1 e 2 respectivamente, como mostrado na Figura 3.5(b). Assim, por exemplo, a posição do trem 1 cresce de D a B e decresce de B a D. As posições dos sensores medidas a partir das origens (pontos D e G) são as seguintes: $A = 20\,\mathrm{km}$, $B = 40\,\mathrm{km}$, $C = 10\,\mathrm{km}$ *e* $D = 0\,\mathrm{km}$ *para o trem 1; e* $E = 30\,\mathrm{km}$, $F = 50\,\mathrm{km}$ *e* $G = 0\,\mathrm{km}$ *para o trem 2.*

Sejam $x_1(\cdot)$ *e* $x_2(\cdot)$ *sinais contínuos com as medidas das trajetórias dos trens 1 e 2, respectivamente. A trajetória de estados em tempo contínuo* $x(\cdot)$ *é um vetor coluna* 2×1 *que armazena os valores de* $x_1(\cdot)$ *e* $x_2(\cdot)$*, ou seja,* $x(\cdot) = [\ x_1(\cdot) \quad x_2(\cdot)\ ]^T$*. Indexam-se as dinâmicas possíveis para* x_1 *e* x_2 *pelos sinais-condição* $u_1(\cdot)$ *e* $u_2(\cdot)$ *que assumem valores em* $U_1 = U_2 = \{1, 2, 3, 4\}$ *conforme a equação (3.2), onde* $i = 1; 2$*:*

$$\dot{x}_i(t) = \begin{cases} -d\ , & se \quad u_i(t) = 1 \\ -r\ , & se \quad u_i(t) = 2 \\ d\ , & se \quad u_i(t) = 3 \\ r\ , & se \quad u_i(t) = 4 \end{cases} \tag{3.2}$$

As dinâmicas para o sinal $x(\cdot)$ *são indexadas pelas condições* $u(t) \in U = \{1, 2, \ldots, 16\}$*, como mostrado na Tabela 3.1.*

Tabela 3.1: Dinâmicas para os trens.

$\dot{x}(t)$	$u_1(t)$	$u_2(t)$	$u(t)$
$[\ -d \quad -d\ ]^T$	1	1	1
$[\ -d \quad -r\ ]^T$	1	2	2
$[\ -d \quad d\ ]^T$	1	3	3
$[\ -d \quad r\ ]^T$	1	4	4
$[\ -r \quad -d\ ]^T$	2	1	5
.	.	.	.
.	.	.	.
$[\ r \quad r\ ]^T$	4	4	16

O cruzamento de um trem por um sensor de posição é interpretado como a ocorrência de um evento de limiar. Assim, cada sensor de posição está associado a uma superfície de limiar.

A função $y(t) = g(x(t))$ *é mostrada na equação (3.3).*

$$y(t) = \begin{bmatrix} x_1(t) - 20 \\ x_1(t) - 40 \\ 10 - x_1(t) \\ -x_1(t) \\ x_2(t) - 30 \\ x_2(t) - 50 \\ -x_2(t) \end{bmatrix} \tag{3.3}$$

No intuito de simplificar, considera-se que apenas um sensor de posição é cruzado a cada instante e então tem-se um total de 7 *eventos de limiar. Além disso, utilizam-se símbolos para representar os eventos de limiar. O símbolo A, por exemplo, representa o evento* $v(t) = [\ 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0\ ]^T$*, o qual ocorre naqueles instantes em que o trem 1 cruza pela posição A, ou seja, quando o trem 1 percorre a distância de* $20\,\mathrm{km}$ *contada a partir da origem* $D = 0$*. Note que neste trajeto o trem 1 percorreu a distância de* $10\,\mathrm{km}$*, fazendo com que em um certo instante* t' *o patamar* T_3 *fosse alcançado (*$y_3(t') = 0$*), mas sem no entanto gerar um evento C, pois neste caso* $(\nexists\Delta > 0)(\forall\delta \in (0, \Delta)) : y_3(t' - \delta) < 0$*, ou seja, o patamar foi cruzado no sentido inverso ao definido para a geração do evento.*

A dinâmica discreta dos trens pode ser expressa por intermédio de autômatos de Moore (Moore, 1956; Hopcroft et al., 2001) [2] *mostrados na Figura 3.6, nos quais as saídas dos estados (números existentes dentro dos estados) são as condições correspondentes da equação (3.2).*

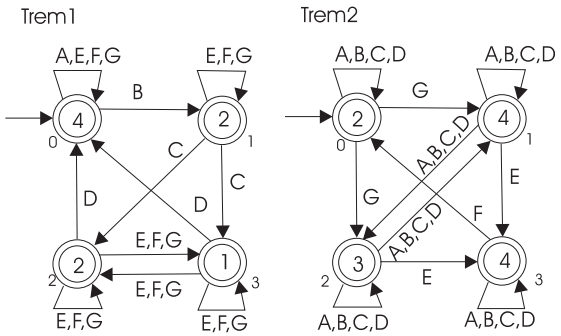


Figura 3.6: Dinâmica discreta para os trens

[2] *Também chamados de Máquinas de Moore na literatura.*

Assim, no estado inicial o autômato para o trem 1 tem como saída a condição $u_1(t) = 4$*, que equivale a escolha da dinâmica* $\dot{x}_1(t) = r$*. Os autômatos mostrados na Figura 3.6 expressam três aspectos: a mudança de sinal da velocidade para o crescimento e decrescimento da variável posição; a forma de chaveamento da velocidade: não determinística, em função da ocorrência de eventos de cruzamento; e a limitação de velocidade devagar aos trechos* $\overline{CD}$ *e* $\overline{GE}$*.* ■

Exemplo 3.2 ***: Sistema de Nível de Líquido***

Seja o sistema de nível de líquido mostrado na Figura 3.7. Para controlar o nível de líquido no tanque (variável $x(t)$*) pode-se atuar sobre uma válvula que regula a entrada de líquido no mesmo. Pode-se então escolher entre abrir completamente a válvula, o que é feito pela aplicação de um sinal de controle* u_{on}*, ou fechar esta válvula por intermédio da aplicação de um sinal de controle* u_{off}*. Sendo assim, o sinal condição de entrada* $u(t)$ *assume valores sobre o conjunto discreto* $U = \{u_{on}, u_{off}\}$*. Assim como no exemplo anterior, neste sistema, mudanças no sinal aplicado à válvula de entrada podem ocorrer apenas nos instantes em que há a sinalização de um evento.*

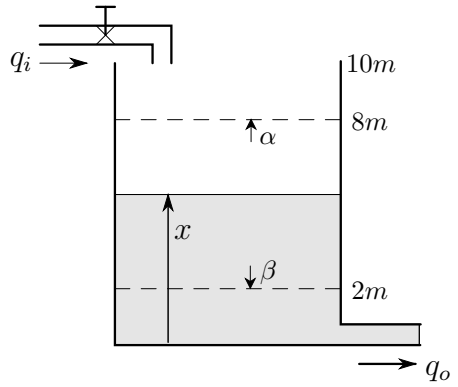


Figura 3.7: Sistema de nível de líquido.

Durante o enchimento do tanque, quando o nível atinge 8 metros é gerado o evento α*. Já durante o esvaziamento do mesmo, quando é atingido o patamar de* $2\,\mathrm{m}$ *é gerado o evento* β*. Note que nenhum evento deve ser gerado quando o patamar de* $8\,\mathrm{m}$ *é alcançado no esvaziamento do tanque, ou quando o patamar de* $2\,\mathrm{m}$ *é atingido durante o seu enchimento.*

Observe que quando o nível atinge $10\,\mathrm{m}$ *(altura do tanque) e a válvula de entrada é mantida aberta o tanque transborda, não havendo mais variação no nível, ou seja,* $\dot{x}(t) = 0$*. Da mesma*

forma, quando o tanque se esvazia e a válvula é mantida fechada, o nível permanece em $0\,\mathrm{m}$ *e* $\dot{x}(t) = 0$.

Na modelagem deste sistema, considera-se que o fluxo (vazão) de saída de líquido no tanque é turbulento e então a relação entre o nível, x*, e a vazão de saída,* q_o*, é dada por* $q_o = K\sqrt{x}$*, onde* $K > 0$ *é uma constante conhecida (Fox e McDonald, 2001; Brunetti, 2004). Considera-se ainda que quando a válvula de entrada está aberta, a vazão de entrada é maior que a vazão de saída, isto é,* $q_i > q_o$*, e assim* $q_i > K\sqrt{x}$ *para qualquer* $0 \leq x \leq 10$*. Desta forma, sempre que a válvula de entrada é aberta ocorre o enchimento do tanque e sempre que a válvula é fechada o tanque se esvazia. Quando a válvula está aberta, a vazão de entrada é constante,* $q_i \neq 0$*, e conhecida.*

A dinâmica contínua da planta pode ser expressa pela seguinte equação:

$$\dot{x} = \begin{cases} \frac{1}{A}(q_i - K\sqrt{x}) & se \quad u = u_{on} \;\; e \;\; 0 < x < 10 \\ -\frac{K}{A}\sqrt{x} & se \quad u = u_{off} \;\; e \;\; 0 < x < 10 \\ 0 & se \quad u = u_{on} \;\; e \;\; x = 10 \quad ou\ se \quad u = u_{off} \;\; e \;\; x = 0 \end{cases}$$

onde A *é a área da seção transversal do tanque.*

Considera-se que inicialmente o tanque está vazio e que a válvula de entrada está aberta, ou seja, $x(0) = 0$ *e* $u(0) = u_{on}$.

A função $y(t) = g(x(t))$ *é expressa como* $y(t) = [\ (x(t) - 8) \quad (2 - x(t))\ ]^T$*. Note então que os símbolos* α *e* β *foram utilizados para representar* $v(t) = [\ 1 \quad 0\ ]^T$ *e* $v(t) = [\ 0 \quad 1\ ]^T$*, respectivamente. Assim, o bloco* detector de zeros *gera o evento* α *quando a superfície* $g_1(x) = x(t) - 8 = 0$ *é alcançada durante o enchimento do tanque, momento em que ocorre uma descontinuidade em* $v_1(t)$*. Já o evento* β *é gerado pelo detector de zeros quando a superfície* $g_2(x) = 2 - x(t) = 0$ *é atingida no esvaziamento do tanque, ou seja, quando ocorre uma descontinuidade em* $v_2(t)$.

Por fim, a dinâmica discreta da planta pode ser expressa pelo autômato de Moore mostrado na Figura 3.8.

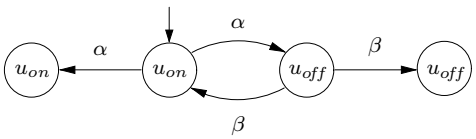


Figura 3.8: Dinâmica discreta para o sistema de nível de líquido.

A Figura 3.9 ilustra um dos possíveis comportamentos para o sistema de nível de líquido.

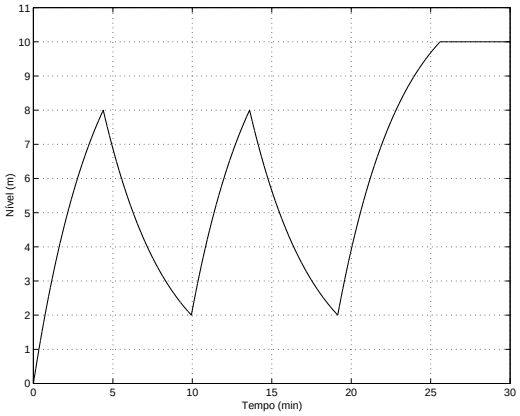


Figura 3.9: Comportamento possível para o sistema de nível.

■

Modelo de Estado Discreto para a Planta Híbrida

Descreve-se o comportamento discreto do sistema condição/evento $\mathcal{H}_c$ pelo seu *modelo de estado discreto* (Sreenivas e Krogh, 1991). Para tanto, introduz-se a *representação de traço discreto* para $\mathcal{H}_c$ como uma quádrupla (W, β, ρ, W_0), descrita como segue.

Considere o sinal condição $w(\cdot)$, constante por partes, contínuo à direita e com limites à esquerda, que registra o valor correspondente da trajetória de estados $x(\cdot)$ apenas nos instantes de descontinuidades em $(v(\cdot), u(\cdot)) \in \mathcal{H}_c$. O conjunto W é o conjunto de todos os valores que o sinal $w(\cdot)$ pode assumir e W_0 é o conjunto dos possíveis valores iniciais para $w(\cdot)$. A diferença entre $w(\cdot)$ e $x(\cdot)$ é que o primeiro não registra a trajetória contínua entre as descontinuidades na entrada e saída do sistema. Assim, W é o mesmo que o conjunto original de estados X, ou seja, $W = \mathbb{R}^n$, e $W_0 = X_0$. A função de transição $\beta : W \times U \to W$ para $w(\cdot)$ possui a forma $w(t) = \beta(w(t^-), u(t^-))$ e é tal que:

$$w(t) = \begin{cases} \Phi_{u(t^-)}(t, w(t^-)) & \text{se para algum } i = 1, 2, .., m,\ g_i(\Phi_{u(t^-)}(t, w(t^-))) = 0 \\ & \bigwedge(\exists \Delta > 0)(\forall \delta \in (0, \Delta)) : g_i(\Phi_{u(t^-)}(t - \delta, w(t^-))) < 0 \\ w(t^-) & \text{de outra forma} \end{cases}$$

onde $\Phi_u(t, x(t_0))$ é a solução da equação diferencial $\dot{x}(t) = f_{u(t)}(x(t))$ para $u(t) \in U$, $t \geq t_0$ e valor inicial $x(t_0)$. Utiliza-se a notação t^- para simbolizar o limite pela esquerda ao instante

t. Observe que transições de estado (mudanças no valor de w) ocorrem apenas nos instantes em que uma superfície de limiar é alcançada, nos moldes da equação (3.1). A função de saída evento $\rho : W \times W \to V$ gera o evento de limiar correspondente no instante de transição do estado $w(\cdot)$ e é nula em qualquer outro instante. Ela é definida como:

$$v(t) = \rho(w(t^-), w(t))$$

Este modelo de traço discreto é semelhante ao modelo de estado discreto de Sreenivas e Krogh (1991), exceto pela existência de um conjunto de estados infinito e não enumerável.

O modelo de estado discreto para $\mathcal{H}_d$ é dado pela quádrupla (Q, δ, ϕ, q_0), onde Q é o conjunto discreto de estados, enumerável e possivelmente infinito, $q_0 = q(0^-)$ é o estado inicial, e as funções de transição $\delta : Q \times V \times M \to 2^Q$ e de saída condição $\phi : Q \to U$ são definidas para $t \in [0, \infty)$ por:

$$q(t) \in \delta(q(t^-), v(t), m(t))$$
$$u(t) = \phi(q(t))$$

Note que transições do estado $q(t)$ no subsistema discreto se dão de forma não determinística, e que o sinal de saída $u(\cdot)$ do subsistema discreto $\mathcal{H}_d$ depende apenas de seu estado discreto atual. Deve-se destacar que no formalismo de sistemas C/E, sinais condição apenas habilitam ou inibem transições de estado, sem forçá-las, enquanto que sinais evento forçam tais transições. Desta forma, nos instantes t de descontinuidade nos sinais evento $v(\cdot)$ e $m(\cdot)$ ocorre uma transição de estado em $\mathcal{H}_d$ e o conjunto de estados $q(t)$ que podem ser alcançados é restrito pelo sinal de controle $m(t)$. A função de transição de estado para $\mathcal{H}_d$ é escrita como $\delta(q(t^-), v(t), m(t)) = \{q(t) \in \delta(q(t^-), v(t), U) : \phi(q(t)) \in m(t)\}$, onde $\delta(q(t^-), v(t), U)$ é a função de transição de estado de $\mathcal{H}_d$ para a planta híbrida em malha aberta. Assim, o conjunto das possíveis condições de entrada a serem aplicadas a $\mathcal{H}_c$ fica restrito a $\bigcup_{q(t) \in \delta(q(t^-), v(t), m(t))} \phi(q(t))$.

Supervisor

O supervisor $\mathcal{F}$ é definido como um subconjunto de $\mathcal{V} \otimes \mathcal{M} \otimes \mathcal{U}$ sendo que para $(v(\cdot), m(\cdot), u(\cdot)), (v'(\cdot), m'(\cdot), u'(\cdot)) \in \mathcal{F}$, tem-se que:

$$((v(t), u(t)) = (v'(t), u'(t))(\forall t \in [t_0, t_1)) \bigwedge (v(t_1) = v'(t_1))) \Longrightarrow m(t_1) = m'(t_1) \qquad (3.4)$$

O supervisor C/E definido desta forma é um sistema causal (Sreenivas e Krogh, 1991). Assim, de acordo com a equação (3.4), se dois sinais condição $u(\cdot)$ e $u'(\cdot)$ são iguais até um determinado instante de tempo infinitesimalmente menor que t_1 e dois sinais evento $v(\cdot)$ e $v'(\cdot)$ são iguais até o instante t_1, então os sinais evento $m(\cdot)$ e $m'(\cdot)$ gerados pelo supervisor são iguais para todo instante $t \leq t_1$.

Descreve-se o comportamento discreto do supervisor condição/evento $\mathcal{F} \subseteq \mathcal{V} \otimes \mathcal{M} \otimes \mathcal{U}$ pelo seu modelo de estado discreto dado pela quádrupla (Z, ξ, ψ, z_0), onde Z é o conjunto de estados, $z_0 = z(0^-)$ é o estado inicial, e as funções de transição de estado $\xi : Z \times U \times V \to Z$ e de saída evento $\psi : Z \times Z \to M$ são definidas para $t \in [0, \infty)$ como:

$$z(t) = \xi(z(t^-), u(t^-), v(t))$$
$$m(t) = \psi(z(t^-), z(t))$$

Observe que a transição de estado do supervisor é determinística e que é forçada pelo sinal evento $v(\cdot)$. Note ainda que a função de saída depende do estado alcançado nesta transição. Assim, nos instantes de descontinuidade no sinal evento $v(\cdot)$, ocorre uma transição de estado no supervisor, provocando então uma descontinuidade no sinal evento $m(\cdot)$.

Finalmente, o fechamento da malha resulta em um sistema C/E autônomo $\mathcal{F}/\mathcal{H} \subseteq \mathcal{V} \otimes \mathcal{U}$ e seu modelo de estado discreto é obtido pela conexão cascata e realimentação dos modelos do supervisor $\mathcal{F}$ e do sistema híbrido $\mathcal{H}$.

Os supervisores C/E de interesse particular deste trabalho são aqueles que não desabilitam todas as condições para um dado evento. Estes supervisores são chamados de consistentes, conforme definição apresentada a seguir.

Definição 3.1 *Um supervisor* $\mathcal{F} \neq \emptyset$ *é dito consistente para uma planta* $\mathcal{H}$ *se* $\forall (v(\cdot), m(\cdot), u(\cdot)) \in \mathcal{H}$ *tal que* $\exists (v'(\cdot), m'(\cdot), u'(\cdot)) \in \mathcal{F}$ *para o qual* $\forall \tau \in [0, t)$, $(v'(\tau), m'(\tau), u'(\tau)) = (v(\tau), m(\tau), u(\tau))$, *então* $\exists (v''(\cdot), m''(\cdot), u''(\cdot)) \in \mathcal{F}$ *com* $(v''(\tau), m''(\tau), u''(\tau)) = (v(\tau), m(\tau), u(\tau))$ *e* $v''(t) = v(t)$.

Considerando que $\mathcal{F} \subseteq \mathcal{H}$, tem-se que o supervisor conserva a característica da planta híbrida (imposta por $\mathcal{H}_d$) de que para todo instante t de descontinuidade do sinal evento $v(\cdot)$, tem-se que $u(t) \in m(t)$. Sendo assim, um supervisor é consistente se para todo sinal evento $v(\cdot)$ na sua entrada, ele aplica um sinal evento $m(\cdot)$ que habilita o subsistema discreto $\mathcal{H}_d$ a aplicar algum sinal condição $u(\cdot)$ ao subsistema contínuo $\mathcal{H}_c$. Desta forma, um supervisor

consistente pode desabilitar a aplicação de qualquer entrada condição por parte de $\mathcal{H}_d$, mas não pode desabilitar todas as possíveis condições de entrada para um dado evento.

Antes de enunciar o problema de controle supervisório para a classe de sistemas híbridos considerada neste trabalho, é necessário expressar o comportamento lógico dos sistemas em termos de linguagens.

Comportamento Lógico

Uma vez que um sistema condição/evento possui sinais de entrada e saída que assumem valores discretos no tempo, é possível associar a este um comportamento lógico, o qual é representado aqui por linguagens de palavras de comprimento finito (Sreenivas e Krogh, 1991).

Seja um sistema condição/evento $\mathcal{D} \subseteq \mathcal{V} \otimes \mathcal{U}$. A linguagem gerada de $\mathcal{D}$, denotada por $\mathcal{L}(\mathcal{D})$, é a linguagem definida sobre palavras de comprimento finito obtidas pelo prefixo-fechamento de todas as cadeias formadas pelas seqüências dos valores que os pares de sinais $(v(\cdot), u(\cdot)) \in \mathcal{V} \otimes \mathcal{U}$ assumem em seus pontos de descontinuidade (sem registrar no entanto os instantes de ocorrência de tais descontinuidades). Vale lembrar que nos instantes de descontinuidade, os sinais $v(\cdot)$ e $u(\cdot)$ assumem valores sobre os conjuntos discretos V e U, respectivamente. Portanto, para o sistema $\mathcal{D}$ tem-se que $\mathcal{L}(\mathcal{D}) \subseteq (V \times U)^*$. Na equação (3.5) apresenta-se uma definição formal da linguagem $\mathcal{L}(\mathcal{D})$, onde o símbolo "$\circ$" denota a concatenação de símbolos $w \in V \times U$.

$$\begin{aligned} \mathcal{L}(\mathcal{D}) = \{ w = w_1 \circ w_2 \circ \ldots \circ w_n \in (V \times U)^* : \exists\, (v(\cdot), u(\cdot)) \in \mathcal{V} \otimes \mathcal{U} \bigwedge \\ t \geq 0 \bigwedge t_1 < t_2 < \ldots < t_n < t, (v(t_i), u(t_i)) = w_i \bigwedge \\ (v(t'), u(t')) = (0, u(t')) \; \forall t' \neq t_1, t_2, \ldots, t_n, t' \leq t \} \end{aligned} \tag{3.5}$$

Conforme visto anteriormente, o valor assumido pela condição $u(t)$ é atualizado somente nos instantes de descontinuidade do sinal evento $v(\cdot)$. Desta forma, para possibilitar a representação da aplicação de uma condição inicial $u(0)$ (antes que tenha ocorrido qualquer evento) associa-se a este um *evento de inicialização* v_0. Portanto, v_0 é um evento fictício que é associado à escolha não determinística do estado inicial $x(0)$ e que não está associado a transições de estado na planta, ou seja, $v_0 \notin V$. O conjunto $V^+ = \{v_0\} \cup V$ denota a inclusão do evento de inicialização em V. Considerando-se ainda o sistema $\mathcal{D}$, tem-se então que $\mathcal{L}(\mathcal{D}) \subseteq (V^+ \times U)^*$, ou mais especificamente $\mathcal{L}(\mathcal{D}) \subseteq (\{v_0\} \times U)(V \times U)^*$, uma vez que eventos v_0 são considerados apenas na inicialização da planta. O exemplo apresentado a seguir ilustra o conceito de linguagem gerada por um sistema C/E.

Exemplo 3.3 *Seja um sistema $\mathcal{D} \subseteq \mathcal{V} \otimes \mathcal{U}$ com uma entrada condição $u(t)$ assumindo valores sobre o conjunto discreto $U = \{u_1, u_2, u_3\}$ e uma saída evento $v(t) = [\ v_1(t) \quad v_2(t)\ ]^T$ assumindo valores não nulos em $V = \{0,1\}^2 - \{0\}^2$. Sejam ainda os símbolos v_1 e v_2 atribuídos a $v(t) = [\ 1 \quad 0\ ]^T$ e $v(t) = [\ 0 \quad 1\ ]^T$, respectivamente. Considere que os sinais $(v(\cdot), u(\cdot)) \in \mathcal{V} \otimes \mathcal{U}$ representados pelas funções mostradas na Figura 3.10 constituem em um possível comportamento temporal para este sistema, ou seja, $(v(\cdot), u(\cdot)) \in \mathcal{D}$.*

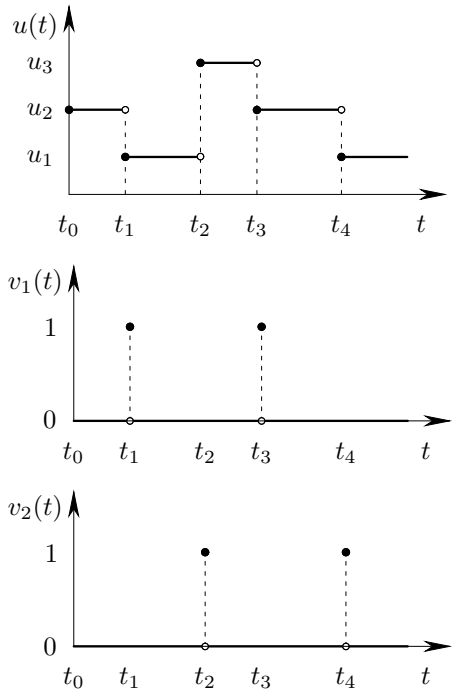


Figura 3.10: Comportamento possível para o sistema $\mathcal{D}$.

A semântica deste comportamento é a seguinte: no instante t_0 em que o sistema começa a ser operado (inicialização) é aplicada uma condição de entrada u_2, o sistema evolui e no instante t_1 atinge uma superfície de limiar, ocorrendo então uma descontinuidade em $v_1(t)$ (evento v_1); neste instante é aplicada a condição u_1 e o sistema continua sua evolução até que atinge outra superfície de limiar no instante t_2, quando ocorre uma descontinuidade em $v_2(t)$ (evento v_2) e é aplicada uma condição u_3, dando continuidade à operação do sistema. Sendo assim, a cadeia $w = v_0u_2 \circ v_1u_1 \circ v_2u_3 \circ v_1u_2 \circ v_2u_1$ representa a seqüência mostrada na Figura 3.10, ou seja $w \in \mathcal{L}(\mathcal{D}) \subseteq (\{v_0\} \times U)(V \times U)^*$.

■

No intuito de representar o comportamento lógico da planta híbrida $\mathcal{H}$ em malha aberta, considera-se que $\forall(v(\cdot), m(\cdot), u(\cdot)) \in \mathcal{H}, m(t) = U$ para todo $t \in [0, \infty)$ tal que $v(t) \neq 0$, ou seja, é como se existisse um sinal de controle $m(\cdot)$ a habilitar toda a gama de condições que $\mathcal{H}_d$ pode aplicar a $\mathcal{H}_c$. Assim, sem perda de generalidade, a linguagem da planta híbrida em malha aberta pode ser escrita como $\mathcal{L}(\mathcal{H}) \subseteq (V^+ \times U)^*$.

Deste ponto em diante, no domínio de SEDs, os eventos de limiar $v(t) \in V$ serão chamados simplesmente de eventos v, as descontinuidades no sinal $m(\cdot)$ serão chamadas de controles m, e as condições $u(t) \in U$ serão chamadas de condições u. Assim, a semântica de uma palavra $\sigma = vu \in \mathcal{L}(\mathcal{H})$ é a seguinte: quando uma variável (ou combinação de variáveis) do espaço de estados contínuo do sistema híbrido $\mathcal{H}$ atinge uma superfície de limiar, o subsistema contínuo gera um evento v e, em resposta, o subsistema discreto aplica uma condição u ao subsistema contínuo.

Ao longo deste trabalho são tratadas três classes de linguagens, a saber: linguagens em V^*, $(V^+ \times U)^*$ e $(V^+ \times M \times U)^*$. A seguir comenta-se sobre a utilização de cada uma destas linguagens.

- As especificações sobre os comportamentos desejados para o sistema híbrido sob supervisão (em malha fechada), são expressas por linguagens $E \subseteq V^*$, ou seja, em termos de seqüências de eventos de limiar. Note que E expressa as propriedades desejadas em temos de seqüências de superfícies de limiar atingidas no espaço de estados contínuo da planta híbrida. Assim, o evento de inicialização v_0 e as condições u sob as quais os eventos v são gerados não fazem parte da especificação.

- Para descrever o comportamento lógico da planta híbrida em malha aberta ($\mathcal{L}(\mathcal{H})$) são utilizadas linguagens em $(V^+ \times U)^*$.

- Linguagens em $(V^+ \times M \times U)^*$ são utilizadas para representar o comportamento do supervisor ($\mathcal{L}(\mathcal{F})$). A semântica de uma palavra $\tau = vmu \in V^+ \times M \times U$ é a seguinte: o supervisor verifica a geração de um evento v e então aplica uma entrada de controle m ao subsistema discreto, que por sua vez aplica uma entrada u ao subsistema contínuo.

Seja a projeção $P_{V\times U} : (V^+\times M\times U)^* \rightarrow (V^+ \times U)^*$ que apaga os símbolos $m \in M$ das palavras em $(V^+\times M\times U)^*$, definida recursivamente por:

$$P_{V\times U}(\epsilon) = \epsilon \ ;$$

$$P_{V\times U}(\tau) = \begin{cases} v_0 u \quad \text{para} \quad \tau = v_0 m u \in (\{v_0\} \times M \times U) \\ vu \quad \text{para} \quad \tau = vmu \in (V \times M \times U) \ ; \ e \end{cases}$$

$$P_{V\times U}(\varpi\circ\tau) = P_{V\times U}(\varpi)\circ P_{V\times U}(\tau) \ \text{ para } \ \varpi \in (V^+\times M\times U)^* \ \text{ e } \ \tau \in (V^+\times M\times U).$$

Esta projeção pode ser naturalmente estendida para linguagens em $(V^+\times M\times U)^*$. Assim, a projeção de uma linguagem $\mathcal{L}(\mathcal{F}) \in (V^+\times M\times U)^*$ em $(V^+\times U)^*$ é definida como:

$$P_{V\times U}[\mathcal{L}(\mathcal{F})] = \{s \in (V^+\times U)^* : (\exists \varpi \in \mathcal{L}(\mathcal{F}))\ P_{V\times U}(\varpi) = s\}.$$

A linguagem da planta $\mathcal{H}$ sob a ação do supervisor $\mathcal{F}$ é definida como $\mathcal{L}(\mathcal{F}/\mathcal{H}) = P_{V\times U}[\mathcal{L}(\mathcal{F})] \cap \mathcal{L}(\mathcal{H})$.

Observe que o comportamento lógico do sistema híbrido sob a ação do supervisor é dado pela linguagem $\mathcal{L}(\mathcal{F}/\mathcal{H}) \subseteq (V^+\times U)^*$, mas que as especificações sobre os comportamentos lógicos desejados para o sistema híbrido em malha fechada são feitas apenas em termos de eventos, ou seja, por linguagens $E \subseteq V^*$. Sendo assim, para que se possa relacionar o comportamento real com o desejado para o sistema sob supervisão, define-se recursivamente a projeção $P_V : (V^+\times U)^* \rightarrow V^*$ como:

$$P_V(\epsilon) = \epsilon \ ;$$

$$P_V(\sigma) = \begin{cases} \epsilon \quad \text{para} \quad \sigma = v_0 u \in \{v_0\} \times U \\ v \quad \text{para} \quad \sigma = vu \in V \times U \ ; \ e \end{cases}$$

$$P_V(s \circ \sigma) = P_V(s) \circ P_V(\sigma) \ \text{ para } \ s \in (V^+ \times U)^* \ \text{ e } \ \sigma \in V^+ \times U \ .$$

A projeção definida anteriormente pode ser estendida de forma natural para linguagens. Assim, a projeção de uma linguagem $K \subseteq (V^+\times U)^*$ em V^* é definida como:

$$P_V(K) = \{t \in V^* : (\exists s \in K) \ \ P_V(s) = t\}.$$

Desta forma, a projeção $P_V : (V^+ \times U)^* \rightarrow V^*$ apaga os símbolos v_0 e os símbolos $u \in U$ de palavras em $(V^+ \times U)^*$.

Ainda, a projeção inversa é dada por $P_V^{-1}(E) = \{s \in (\{v_0\} \times U)(V \times U)^* : P_V(s) \in E\}$.

Definição 3.2 *Um supervisor $\mathcal{F}$, consistente para $\mathcal{H}$, é dito ser um supervisor C/E marcador em relação a E se $\mathcal{L}_m(\mathcal{F}/\mathcal{H}) = \mathcal{L}(\mathcal{F}/\mathcal{H}) \cap P_V^{-1}(E)$.*

A linguagem $\mathcal{L}_m(\mathcal{F}/\mathcal{H})$ é interpretada como o conjunto das cadeias $w \in (V^+ \times U)^*$ formadas pelas seqüências dos valores que os pares de sinais $(v(\cdot), u(\cdot)) \in \mathcal{V} \otimes \mathcal{U}$ assumem em seus pontos de descontinuidade e tais que $P_V(w)$ representa uma seqüência de eventos de limiar que corresponde a finalização de uma tarefa por parte do sistema híbrido sob supervisão. Assim, a projeção $P_V[\mathcal{L}_m(\mathcal{F}/\mathcal{H})] \subseteq V^*$ resulta no conjunto das seqüências de eventos de limiar que correspondem à realização de tarefas no sistema híbrido sob supervisão. Por fim, pode-se afirmar que $\mathcal{L}_m(\mathcal{F}/\mathcal{H}) \subseteq \mathcal{L}(\mathcal{F}/\mathcal{H}) \subseteq (V^+ \times U)^*$.

Na seqüência, introduz-se algumas definições necessárias para que se possa apresentar o problema de síntese de supervisores para sistemas híbridos.

Seja uma linguagem $K \subseteq (V^+ \times U)^*$. O prefixo-fechamento de K é dado por:

$$\overline{K} = \{s \in (V^+ \times U)^* : (\exists\, w \in (V^+ \times U)^*\ s \circ w \in K\}.$$

K é dita ser *prefixo-fechada* se $K = \overline{K}$. Seja agora uma linguagem $E \subseteq V^*$. O prefixo-fechamento de E é dado por:

$$\overline{E} = \{r \in V^* : (\exists\, t \in V^*)\ r \circ t \in E\}.$$

E é dita ser *prefixo-fechada* se $E = \overline{E}$.

Outra definição importante é a de supervisor C/E marcador não bloqueante, conforme segue.

Definição 3.3 *Seja $\mathcal{F}$ um supervisor marcador em relação a uma especificação E. $\mathcal{F}$ é dito ser um supervisor C/E marcador não bloqueante (SCEMN) se $\overline{\mathcal{L}_m(\mathcal{F}/\mathcal{H})} = \mathcal{L}(\mathcal{F}/\mathcal{H})$.*

No intuito de abreviar a notação, no restante deste trabalho omitir-se-á a expressão *"em relação a E"* sempre que a especificação E estiver implícita (sempre que não houver dúvida quanto a especificação considerada).

Pode-se apresentar agora o problema de síntese de supervisores para sistemas híbridos.

Problema 3.1 (Síntese de Supervisores para Sistemas Híbridos – SSSH) *Seja $\mathcal{H}$ o modelo C/E da planta híbrida e dada a especificação $E \subseteq V^*$ para o comportamento da planta em malha fechada, encontrar um SCEMN $\mathcal{F}$ tal que:*

$$\emptyset \neq P_V[\mathcal{L}_m(\mathcal{F}/\mathcal{H})] \subseteq E.$$

Na próxima seção utiliza-se a teoria de controle supervisório de SEDs para propor uma solução formal para o problema apresentado acima.

3.3 Abordagem por Controle de Sistemas a Eventos Discretos

Nesta seção, o problema de síntese de supervisores para sistemas híbridos (SSSH), anteriormente formulado, é traduzido para uma abordagem de controle puramente discreta. Mostra-se então que este problema pode ser solucionado por intermédio de um *problema equivalente no domínio de SEDs.*

Antes de apresentar um modelo de sistemas a eventos discretos para o sistema híbrido é necessário definir dois novos conjuntos. O conjunto ativo de eventos em $K \subseteq (V^+ \times U)^*$ após $w \in \overline{K}$ é dado por $V_K(w) = \{v \in V^+ : (\exists u \in U)\ w \circ vu \in \overline{K}\}$. O conjunto ativo de condições em K após $w \in \overline{K}$ e para um dado $v \in V_K(w)$ é definido como $U_K(w, v) = \{u \in U : w \circ vu \in \overline{K}\}$.

O modelo de sistemas a eventos discretos com estrutura de controle para a planta híbrida $\mathcal{H}$ é a dupla $H = (L, \Gamma)$, onde $L = \mathcal{L}(\mathcal{H})$, isto é, L é a linguagem gerada pelo sistema híbrido em malha aberta; e a estrutura de controle é um mapa $\Gamma : L \to 2^{2^{V^+ \times U}}$ tal que para todo $w \in L$ tem-se que:

$$\Gamma(w) = \{\gamma \in 2^{V^+ \times U} : (\forall v \in V_L(w))\ (\exists u \in U_L(w, v)) : vu \in \gamma\}.$$

Observe que a estrutura de controle depende da palavra gerada pelo sistema. Conforme definida, a estrutura de controle traduz a idéia de que para um dado evento v, ativo após uma cadeia $w \in L$, deve sempre haver pelo menos uma condição u habilitada. Note que a inibição de todas as condições possíveis para um dado evento significaria que o supervisor não previu

uma resposta para a ocorrência de uma dada cadeia, não havendo então uma contrapartida física para essa situação.

A seguir, ilustra-se o conceito de Γ por intermédio de um exemplo.

Exemplo 3.4 *Considere que o autômato mostrado na Figura 3.11 represente o modelo de eventos discretos $H = (L, \Gamma)$ para uma planta híbrida $\mathcal{H}$. A linguagem reconhecida por este autômato é dada por $L = \overline{v_1u_1 \circ v_2u_2 \circ (v_3u_1 + v_3u_2 + v_4u_1)} \subseteq (V \times U)^*$. Então, para $w = v_1u_1 \circ v_2u_2 \in L$ tem-se que $\Gamma(w) = \{\{v_3u_1, v_3u_2, v_4u_1\}, \{v_3u_1, v_4u_1\}, \{v_3u_2, v_4u_1\}\} = \{\gamma_1, \gamma_2, \gamma_3\}$. Note que $\gamma = \{v_3u_1, v_3u_2\} \notin \Gamma(w)$ uma vez que não contempla o evento $v_4 \in V_L(w)$.*

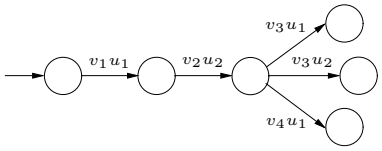


Figura 3.11: Autômato para ilustração de Γ.

■

O supervisor de eventos discretos F para a planta $H = (L, \Gamma)$ é definido pelo mapa $F : L \to 2^{V^+ \times U}$. Escreve-se F/H para denotar que a planta H está sob supervisão de F.

O supervisor pode ser representado por um autômato de estados finitos $F = (P, V^+ \times U, \theta, p_0)$, onde P é o conjunto de estados, p_0 é o estado inicial, e $\theta : P \times (V^+ \times U) \to P$ é a função de transição de estado.

Exemplo 3.5 *Considere que o autômato mostrado na Figura 3.12 representa um supervisor F_1 para a planta mostrada na Figura 3.11. Assim, para $w = v_1u_1 \circ v_2u_2 \in L$ tem-se que $F_1(w) = \{v_3u_1, v_3u_2\}$.*

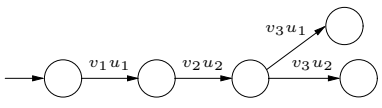


Figura 3.12: Supervisor F_1.

■

Observe que nenhuma restrição é feita no sentido de $F(w) \in \Gamma(w)$. Sendo assim, para um dado evento v, ativo após uma cadeia $w \in L$, pode ser que o supervisor não habilite

nenhuma condição u. Este problema é tratado mais adiante, quando introduz-se a definição de *supervisor consistente*.

A linguagem $L(F/H) \subseteq L$ representa o comportamento gerado da planta $H = (L, \Gamma)$ sob a ação de um supervisor F, e é definida recursivamente como:

1. $\epsilon \in L(F/H)$; e
2. $w \circ vu \in L(F/H) \Longleftrightarrow w \in L(F/H) \bigwedge w \circ vu \in L \bigwedge vu \in F(w)$.

Na seqüência, apresenta-se a definição de supervisor consistente no domínio de SEDs.

Definição 3.4 *Um supervisor F é dito consistente para uma planta $H = (L, \Gamma)$ se:*

$$(\forall w \in L(F/H))\ F(w) \in \Gamma(w).$$

Assim, um supervisor é *consistente* para uma planta se para qualquer cadeia $w \in L(F/H)$, e para todo evento v possível de ocorrer na planta após w, existe pelo menos uma condição de entrada u habilitada pelo supervisor, tal que $vu \in \gamma \in \Gamma(w)$. O conceito de consistência de supervisores é ilustrado por intermédio do exemplo que segue.

Exemplo 3.6 *Considere que o supervisor F_1, mostrado na Figura 3.12, seja utilizado para controlar a planta da Figura 3.11. O supervisor F_1 não é consistente para esta planta uma vez que para $w = v_1u_1 \circ v_2u_2 \in L(F_1/H)$ tem-se que $F_1(w) = \{v_3u_1, v_3u_2\} \notin \Gamma(w)$. Observe que se após a cadeia $w \in L$ ocorrer o evento $v_4 \in V_L(w)$, o supervisor não habilitará nenhuma condição, não tendo portanto uma resposta para esta cadeia. Um supervisor consistente para aquela planta é mostrado na Figura 3.13. Note agora que $F_2(w) = \{v_3u_1, v_4u_1\} \in \Gamma(w)$.*

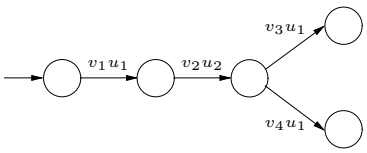


Figura 3.13: Supervisor F_2.

■

Definição 3.5 *Um supervisor F, consistente para H, é dito ser um supervisor marcador em relação a E se $L_m(F/H) = L(F/H) \cap P_V^{-1}(E)$.*

A seguir, introduz-se a definição de *supervisor marcador não bloqueante* para o domínio de SEDs. Esta definição é análoga àquela apresentada por intermédio da Definição 3.3, no domínio de sistemas C/E.

Definição 3.6 *Seja F um supervisor marcador em relação a uma especificação E. F é dito ser um supervisor marcador não bloqueante (SMN) se $\overline{L_m(F/H)} = L(F/H)$.*

Assim, um supervisor consistente para uma dada planta é dito ser um supervisor marcador não bloqueante em relação a especificação E se toda cadeia $w \in L(F/H)$ pode ser completada em uma palavra que corresponde a finalização de uma tarefa nesta planta. Observe que a escolha de quais são as tarefas da planta (marcação de estados) é feita por intermédio da especificação E, ou seja, o supervisor é que determina os estados a serem marcados (desmarcados) na planta.

Definição 3.7 *Seja H o modelo de sistemas a eventos discretos para a planta híbrida $\mathcal{H}$. Um supervisor $\mathcal{F}$ para $\mathcal{H}$ é dito ser (logicamente) equivalente a um supervisor F para H se $\mathcal{L}(\mathcal{F}/\mathcal{H}) = L(F/H)$ e $\mathcal{L}_m(\mathcal{F}/\mathcal{H}) = L_m(F/H)$.*

Em (González et al., 2001) foi introduzido um procedimento formal para a obtenção de um supervisor condição/evento $\mathcal{F}$, conforme definido anteriormente, e de forma que $\mathcal{F}$ seja logicamente equivalente ao supervisor de eventos discretos F. Assim, o supervisor condição/evento definido como $\mathcal{F} \subseteq \mathcal{V} \otimes \mathcal{M} \otimes \mathcal{U}$ é tal que:

$$(\forall \varpi \in \mathcal{L}(\mathcal{F}))(\forall vmu \in V^+ \times M \times U)\ ,\ \varpi \circ vmu \in \mathcal{L}(\mathcal{F}) \Longleftrightarrow vu \in F(P_{V\times U}(\varpi)),$$

isto é, as ações de controle dos supervisores são equivalentes.

Proposição 3.1 *(González et al., 2001) O supervisor C/E $\mathcal{F}$ obtido segundo o Algoritmo 1 em (González et al., 2001) é logicamente equivalente ao supervisor SED F usado como base.*

A Proposição 3.1 indica que o problema de síntese de supervisores para sistemas híbridos (Problema 3.1 apresentado na seção anterior) pode ser solucionado pela resolução de um problema de controle supervisório equivalente no domínio de SEDs. Tal problema equivalente é enunciado a seguir.

Problema 3.2 (Problema de Controle Supervisório Equivalente – PCSE) *Seja* $H = (L, \Gamma)$ *o modelo de sistemas a eventos discretos para a planta híbrida* $\mathcal{H}$ *e dada a especificação* $E \subseteq V^*$ *para o comportamento da planta em malha fechada, encontrar um SMN* F *tal que:*

$$\emptyset \neq P_V[L_m(F/H)] \subseteq E.$$

Um conceito chave para a solução de qualquer problema de controle supervisório é o de controlabilidade de linguagens. A definição apresentada a seguir consiste em uma adaptação ao nosso contexto da definição introduzida em (González, 2000).

Definição 3.8 *Sejam as linguagens* $K \subseteq L \subseteq (V^+ \times U)^*$. K *é dita ser vu-controlável e.r.a. (em relação a)* L *se:*

$$(\forall w \in \overline{K})\ \ V_L(w) = V_K(w). \tag{3.6}$$

Assim, K é *vu-controlável* e.r.a. L se após qualquer cadeia $w \in \overline{K}$, todos e somente aqueles eventos possíveis de ocorrer em $\overline{K}$ podem ocorrer também em L.

Seja $\Sigma_K(w) = \{vu \in V^+ \times U : w \circ vu \in \overline{K}\}$ o conjunto ativo de condições/eventos em K após $w \in \overline{K}$. Seja ainda um modelo SED com estrutura de controle $H = (L, \Gamma)$. A condição apresentada na equação (3.6) equivale à seguinte condição:

$$(\forall w \in \overline{K})(\exists \gamma \in \Gamma(w)) : \gamma \cap \Sigma_L(w) = \Sigma_K(w).$$

Ou seja, uma linguagem K é *vu-controlável* e.r.a. L se para toda cadeia $w \in \overline{K}$ existe pelo menos um padrão de controle $\gamma \in \Gamma(w)$ que habilite na planta o mesmo conjunto ativo de condições/eventos habilitados em K após w.

Diferentemente do que acontece na teoria de controle supervisório de SEDs (abordagem de Ramadge–Wonham), para os sistemas condição/evento não se divide o alfabeto em eventos controláveis e não controláveis. Na abordagem de sistemas C/E os eventos não podem ser desabilitados e o controle das seqüências de eventos é feito pela escolha apropriada para as condições associadas aos eventos.

Retornando ao paradigma de sistemas condição/evento, da mesma forma que no domínio das linguagens em $(V^+ \times U)^*$, podemos introduzir o conceito de controlabilidade para linguagens em V^*, conforme feito a seguir.

Definição 3.9 *Sejam as linguagens $L \subseteq (V^+ \times U)^*$ e $E \subseteq P_V(L)$. E é dita ser v-controlável e.r.a. L se $\exists K \subseteq L$ vu-controlável e.r.a. L, tal que $P_V(K) = E$.*

Sejam, por exemplo, as linguagens $E_1 = v_1 \circ v_2$ e $L = [\overline{v_1u_2 \circ (v_3u_1 + v_2u_2)}] + [\overline{v_1u_1 \circ (v_2u_1 + v_2u_2 \circ v_3u_2)}]$. E_1 é v-controlável e.r.a. L uma vez que existe uma linguagem vu-controlável $K_1 = v_1u_1 \circ v_2u_1$ tal que $P_V(K_1) = E_1$. Note que a v-controlabilidade de uma linguagem em V^* está diretamente associada à definição de vu-controlabilidade apresentada anteriormente, o que está de acordo com o esperado, uma vez que o controle é feito por intermédio dos sinais condição. Sendo assim, a análise sobre a controlabilidade se dá, de fato, no domínio $(V^+ \times U)^*$.

Observe que se $E \nsubseteq P_V(L)$, sendo $E = P_V(K)$, tem-se que $P_V(K) \nsubseteq P_V(L)$ e então $K \nsubseteq L$. Sendo assim, K não é vu-controlável e.r.a. L, e portanto E não é v-controlável e.r.a. L.

Muitas vezes, pode ser interessante escrever a especificação para o comportamento do sistema em malha fechada como uma linguagem $K \subseteq (V^+ \times U)^*$. Neste caso, pode-se referir ao supervisor como sendo um SMN em relação a K. Assim, para um SMN F em relação a K tem-se que $L_m(F/H) = L(F/H) \cap K$. Novamente, deste ponto em diante, omitir-se-á a expressão *"em relação a K"* sempre que a especificação K estiver implícita.

A seguinte proposição apresenta as condições necessárias e suficientes para a existência de um supervisor que garanta o cumprimento de uma especificação $K \subseteq (V^+ \times U)^*$.

Proposição 3.2 *Seja uma planta $H = (L, \Gamma)$ e uma especificação $K \subseteq (V^+ \times U)^*$, $K \neq \emptyset$. Existe um SMN F para H tal que $L_m(F/H) = K$ se e somente se K é vu-controlável e.r.a. L.*

Prova:

(Se) *Inicialmente prova-se que $L(F/H) = \overline{K}$.*

1. *$L(F/H) \subseteq \overline{K}$.*

   Seja F um supervisor consistente para H e seja $w \circ vu \in L(F/H)$. Então, de acordo com a definição de $L(F/H)$ tem-se que $w \in L(F/H) \bigwedge w \circ vu \in L \bigwedge vu \in F(w)$. Pela definição de supervisor consistente sabe-se que para $w \in L(F/H)$ tem-se que $F(w) \in \Gamma(w)$, logo $vu \in \gamma \in \Gamma(w)$. Note que $w \circ vu \in L(F/H)$ implica $w \circ vu \in L$ e então $vu \in \Sigma_L(w)$. Supor agora que K é vu-controlável e.r.a. L. Assumindo, por indução, que $w \in \overline{K}$, e pela definição de vu-controlabilidade, tem-se que $\exists \gamma \in \Gamma(w) : \gamma \cap \Sigma_L(w) = \Sigma_K(w)$. Logo, $vu \in \Sigma_K(w)$, ou seja, $w \circ vu \in \overline{K}$. Portanto, $L(F/H) \subseteq \overline{K}$.

2. $L(F/H) \supseteq \overline{K}$.

   Seja $w \in \overline{K}$ *tal que* $vu \in \Sigma_K(w)$, *ou seja,* $w \circ vu \in \overline{K}$. *Como* $K \subseteq L$, *então* $\overline{K} \subseteq \overline{L} = L$. *Assim,* $w \circ vu \in L$, *o que implica* $vu \in \Sigma_L(w)$. *Supondo que* K *é vu-controlável e.r.a.* L, *tem-se que* $(\forall w \in \overline{K})(\exists \gamma \in \Gamma(w)) : \gamma \cap \Sigma_L(w) = \Sigma_K(w)$. *Logo,* $\exists \gamma \in \Gamma(w)$ *tal que* $vu \in \gamma$. *Seja agora um supervisor consistente* F *para* H *tal que* $vu \in F(w)$. *Assumindo por indução que* $w \in L(F/H)$ *tem-se que* $w \in L(F/H) \bigwedge w \circ vu \in L \bigwedge vu \in F(w)$, *o que implica* $w \circ vu \in L(F/H)$. *Portanto* $L(F/H) \supseteq \overline{K}$.

Sendo assim, $L(F/H) = \overline{K}$.

Continuando a prova, sabe-se que $L_m(F/H) = L(F/H) \cap K$, *mas como* $L(F/H) = \overline{K}$, *então* $L_m(F/H) = \overline{K} \cap K = K$. *Portanto existe um supervisor* F *para* H *tal que* $L_m(F/H) = K$. *Mas* $L_m(F/H) = K$ *implica* $\overline{L_m(F/H)} = \overline{K}$ *e como* $\overline{K} = L(F/H)$, *tem-se que* $\overline{L_m(F/H)} = L(F/H)$, *ou seja,* F *é não bloqueante para* H.

(Somente se)

Seja F *um supervisor marcador não bloqueante para* H *tal que* $L_m(F/H) = K$. *Sabe-se que se* F *é um SMN para* H, *então* F *é consistente para* H *e assim tem-se que:*

$$(\forall w \in L(F/H))\ F(w) \in \Gamma(w). \tag{3.7}$$

Como F *é não bloqueante para* H, *então* $\overline{L_m(F/H)} = L(F/H) = \overline{K}$. *Sabe-se também que* $\Gamma(w) = \{\gamma : (\forall v \in V_L(w))(\exists u \in U_L(w, v)) : vu \in \gamma\}$. *Assim, a equação (3.7) pode ser escrita como:*

$$(\forall w \in \overline{K})\ F(w) = \gamma \in \Gamma(w). \tag{3.8}$$

Note agora que $L(F/H) = \overline{K}$ *implica* $(\forall w \in \overline{K})\ F(w) = \Sigma_K(w)$. *Portanto, pode-se escrever (3.8) na forma:*

$$(\forall w \in \overline{K})(\exists \gamma \in \Gamma(w)) : \gamma = \Sigma_K(w). \tag{3.9}$$

Sabe-se, por fim, que $\overline{K} \subseteq L$ *implica* $(\forall w \in \overline{K})\ \Sigma_K(w) \subseteq \Sigma_L(w)$ *e portanto a equação (3.9) pode ser apresentada na seguinte forma:*

$$(\forall w \in \overline{K})(\exists \gamma \in \Gamma(w)) : \gamma \cap \Sigma_L(w) = \Sigma_K(w).$$

Ou seja, K *é vu-controlável e.r.a.* L. $\diamond$

De forma semelhante, pode-se apresentar uma proposição que trata da existência de um supervisor que garanta o cumprimento de uma especificação $E \subseteq V^*$.

Proposição 3.3 *Seja uma planta $H = (L, \Gamma)$ e uma especificação $E \subseteq V^*$, $E \neq \emptyset$. Existe um SMN F para H tal que $P_V[L_m(F/H)] = E$ se e somente se E é v-controlável e.r.a. L.*

Prova:

(Se) *Pela definição de v-controlabilidade sabe-se que se E é v-controlável e.r.a. L, então existe K vu-controlável e.r.a. L tal que $P_V(K) = E$. Sendo assim, pela Proposição 3.2 pode-se afirmar que existe um SMN F para H tal que $L_m(F/H) = K$ e portanto existe um SMN F para H tal que $P_V[L_m(F/H)] = P_V(K) = E$.*

(Somente se) *Seja F um SMN para H tal que $P_V[L_m(F/H)] = E$. Supor que $L_m(F/H) = K$. Pela Proposição 3.2 pode-se afirmar que K é vu-controlável e.r.a. L. Note ainda que $P_V[L_m(F/H)] = P_V(K) = E$. Pela definição de v-controlabilidade, se existe K vu-controlável tal que $P_V(K) = E$, então E é v-controlável e.r.a. L.*

$\diamond$

Mas e se as linguagens (especificações) K e E não forem controláveis? Neste caso, deve-se determinar o comportamento menos restritivo possível que se pode realizar no sistema e que não fere a especificação, conforme estudado na seqüência.

Seja o conjunto de todas as sublinguagens de K que são *vu*-controláveis em relação a L definido como:

$$\mathbb{C}_{VU}(K) = \{K' \subseteq K : (\forall w \in \overline{K'})V_{K'}(w) = V_L(w)\}.$$

A Proposição 3.4 formaliza a existência de uma (máxima) linguagem *vu*-controlável que contém todas as sublinguagens de K que são *vu*-controláveis em relação a L.

Proposição 3.4 *O conjunto $\mathbb{C}_{VU}(K)$ é não vazio e fechado sob uniões arbitrárias. Em particular, $\mathbb{C}_{VU}(K)$ contém um elemento supremo único, chamado* máxima linguagem *vu*-controlável*, denotado por $Sup\mathbb{C}_{VU}(K)$.*

Prova: *Por definição, $\emptyset$ é vu-controlável em relação a qualquer linguagem L, logo $\emptyset \in \mathbb{C}_{VU}(K)$ e assim $\mathbb{C}_{VU}(K)$ é não vazio.*

Seja $K_i \in \mathbb{C}_{VU}(K)$ para todo i pertencente a um conjunto de índices I, então pode-se afirmar que $(\forall i \in I)(\forall w \in \overline{K_i}), V_{K_i}(w) = V_L(w)$, de onde se obtém que $(\forall w \in$

$\bigcup_{i\in I}\overline{K_i})\bigcup_{i\in I}V_{K_i}(w) = V_L(w)$, *ou ainda que* $(\forall w \in \overline{\cup_{i\in I}K_i})V_{\cup_{i\in I}K_i}(w) = V_L(w)$. *Fazendo* $K' = \bigcup_{i\in I}K_i$ *tem-se que* $(\forall w \in \overline{K'})V_{K'}(w) = V_L(w)$, *ou seja,* K' *é vu-controlável e.r.a.* L *e portanto* $K' \in \mathbb{C}_{VU}(K)$.

Finalmente, $Sup\mathbb{C}_{VU}(K) = \bigcup\{K' : K' \in \mathbb{C}_{VU}(K)\}$. ◇

Com base na Proposição 3.2 pode-se enunciar a seguinte proposição.

Proposição 3.5 *Seja uma planta* $H = (L, \Gamma)$ *e uma especificação* $K \subseteq (V^+ \times U)^*$ *para a qual* $Sup\mathbb{C}_{VU}(K) \neq \emptyset$. *Então, existe um SMN* F *para* H *tal que* $L_m(F/H) = Sup\mathbb{C}_{VU}(K)$.

Prova: *Por definição,* $Sup\mathbb{C}_{VU}(K)$ *é uma linguagem vu-controlável e.r.a.* L. *Assim, pela Proposição 3.2 pode-se afirmar que existe um supervisor marcador* F, *não bloqueante para* H, *tal que* $L_m(F/H) = Sup\mathbb{C}_{VU}(K)$. ◇

Seja agora o conjunto de todas as sublinguagens de E que são v-controláveis em relação a L definido como:

$$\mathbb{C}_V(E) = \{E' \subseteq E : E' \text{ é } v\text{-controlável e.r.a. } L\}.$$

A Proposição 3.6 indica a existência de uma máxima linguagem v-controlável em relação a L.

Proposição 3.6 *O conjunto* $\mathbb{C}_V(E)$ *é não vazio e fechado sob uniões arbitrárias. Em particular,* $\mathbb{C}_V(E)$ *contém um elemento supremo único, chamado* máxima linguagem v-controlável, *denotado por* $Sup\mathbb{C}_V(E)$.

Prova: *Por definição,* $\emptyset$ *é* v*-controlável em relação a qualquer linguagem* L, *logo* $\emptyset \in \mathbb{C}_V(E)$ *e assim* $\mathbb{C}_V(E)$ *é não vazio.*

Seja $E_i \in \mathbb{C}_V(E)$ *para todo* i *pertencente a um conjunto de índices* I, *e seja* $E' = \bigcup_{i\in I}E_i$. *Sabe-se que* $E_i \subseteq E$ *para todo* $i \in I$ *e que portanto* $E' \subseteq E$. *Assim, para provar que* $\mathbb{C}_V(E)$ *é fechado sob uniões arbitrárias, deve-se mostrar que* E' *é* v*-controlável e.r.a.* L, *ou seja, deve-se mostrar que* $\exists K'$ *vu-controlável e.r.a.* L *tal que* $P_V(K') = E'$. *Por definição, sabe-se que para* E_i v*-controlável e.r.a.* L, *tem-se que* $\exists K_i$ *vu-controlável e.r.a.* L *tal que* $P_V(K_i) = E_i$. *Conforme mostrado na Proposição 3.4, se* K_i *é vu-controlável e.r.a.* L *para todo* $i \in I$, *então* $\bigcup_{i\in I}K_i$ *também é vu-controlável e.r.a.* L. *Assim, resta mostrar que* $P_V(\bigcup_{i\in I}K_i) = \bigcup_{i\in I}P_V(K_i) = \bigcup_{i\in I}E_i$.

Sejam $E_1, E_2 \subseteq E$ duas linguagens v-controláveis em relação a L. Pode-se afirmar então que $\exists\, K_1$ vu-controlável : $P_V(K_1) = E_1$ e $\exists\, K_2$ vu-controlável : $P_V(K_2) = E_2$. Mas,

$$\begin{aligned}
P_V(K_1 \cup K_2) &= \{t \in V^* : (\exists w \in (K_1 \cup K_2))\ P_V(w) = t\} \\
&= \{t \in V^* : ((\exists w \in K_1) \quad ou \quad (\exists w \in K_2))\ P_V(w) = t\} \\
&= \{t \in V^* : ((\exists w_1 \in K_1)\ P_V(w_1) = t \quad ou \quad (\exists w_2 \in K_2))\ P_V(w_2) = t\} \\
&= \{t \in V^* : (\exists w_1 \in K_1)\ P_V(w_1) = t\} \cup \{t \in V^* : (\exists w_2 \in K_2)\ P_V(w_2) = t\} \\
&= P_V(K_1) \cup P_V(K_2) \\
&= E_1 \cup E_2.
\end{aligned}$$

Pode-se expandir a demonstração feita acima para um número qualquer de linguagens K_i.

$$\begin{aligned}
P_V(\textstyle\bigcup_{i\in I} K_i) &= \{t \in V^* : (\exists w \in \textstyle\bigcup_{i\in I} K_i)\ P_V(w) = t\} \\
&= \textstyle\bigcup_{i\in I} \{t \in V^* : (\exists w \in K_i)\ P_V(w) = t\} \\
&= \textstyle\bigcup_{i\in I} P_V(K_i) \\
&= \textstyle\bigcup_{i\in I} E_i \\
P_V(K') &= E' .
\end{aligned}$$

Logo, E' é v-controlável em relação a L e portanto $E' \in \mathbb{C}_V(E)$.
Finalmente, $Sup\mathbb{C}_V(E) = \bigcup \{E' : E' \in \mathbb{C}_V(E)\}$. ◇

A proposição enunciada a seguir mostra a relação existente entre as duas máximas linguagens controláveis introduzidas anteriormente.

Proposição 3.7 *Sejam as linguagens $L \subseteq (V^+ \times U)^*$ e $E \subseteq V^*$. Seja ainda $K = P_V^{-1}(E) \cap L$. Se $E \subseteq P_V(L)$ então $Sup\mathbb{C}_V(E) = P_V[Sup\mathbb{C}_{VU}(K)]$. Além disso, $Sup\mathbb{C}_V(E) \neq \emptyset$ se e somente se $Sup\mathbb{C}_{VU}(K) \neq \emptyset$.*

Prova: *Seja $K = P_V^{-1}(E) \cap L$. Inicialmente, mostra-se que se $E \subseteq P_V(L)$ então $P_V(K) = E$.*

Sabe-se que $P_V^{-1}(E) \cap L \subseteq P_V^{-1}(E)$, logo $P_V(K) = P_V[P_V^{-1}(E) \cap L] \subseteq P_V[P_V^{-1}(E)] = E$. Portanto $P_V(K) \subseteq E$. Supor agora que $E \subseteq P_V(L)$, então $\exists L' \subset L$ tal que $P_V(L') = E$. Assim, $P_V^{-1}(E) = P_V^{-1}[P_V(L')] \supseteq L'$ e portanto $P_V^{-1}(E) \cap L \supseteq L'$. Desta forma, $P_V(K) = P_V[P_V^{-1}(E) \cap L] \supseteq P_V(L') = E$ e portanto $P_V(K) \supseteq E$. Sendo assim, tem-se que $P_V(K) = E$.

Continuando, mostra-se que $Sup\mathbb{C}_V(E) = P_V[Sup\mathbb{C}_{VU}(K)]$.

1. $Sup\mathbb{C}_V(E) \supseteq P_V[Sup\mathbb{C}_{VU}(K)]$.

   Supor que $Sup\mathbb{C}_V(E) \not\supseteq P_V[Sup\mathbb{C}_{VU}(K)]$, *ou seja, que* $\exists w \in Sup\mathbb{C}_{VU}(K)$ *tal que* $P_V(w) = t \notin Sup\mathbb{C}_V(E)$. *Seja* $w \in Sup\mathbb{C}_{VU}(K)$, *então* $\{w\}$ *é vu-controlável e.r.a.* L *e, conseqüentemente,* $\{t\}$ *é v-controlável e.r.a.* L. *Sabe-se que* $w \in K$, *logo* $P_V(w) = t \in P_V(K)$. *Mas para* $K = P_V^{-1}(E) \cap L$ *tem-se que* $P_V(K) \subseteq E$, *logo* $P_V(w) = t \in P_V(K) \subseteq E$ *e assim* $t \in E$. *Entretanto, se* $\{t\}$ *é v-controlável e.r.a.* L *e* $t \in E$, *então* $t \in Sup\mathbb{C}_V(E)$, *contrariando a suposição inicialmente apresentada. Sendo assim, tem-se que* $Sup\mathbb{C}_V(E) \supseteq P_V[Sup\mathbb{C}_{VU}(K)]$.

2. $Sup\mathbb{C}_V(E) \subseteq P_V[Sup\mathbb{C}_{VU}(K)]$.

   Seja $t \in Sup\mathbb{C}_V(E)$. *Então* $\{t\}$ *é v-controlável e.r.a.* L *e assim, pela definição de v-controlabilidade, pode-se afirmar que existe* $\{w\}$ *vu-controlável e.r.a.* L, *tal que* $P_V(w) = t$. *Sabe-se que* $Sup\mathbb{C}_V(E) \subseteq E$, *logo* $t \in Sup\mathbb{C}_V(E) \Longrightarrow t \in E$. *Conforme mostrado no início desta prova, para* $K = P_V^{-1}(E) \cap L$, *onde* $E \subseteq P_V(L)$, *tem-se que* $P_V(K) = E$. *Assim, se* $t \in E$ *então* $\exists w \in K$ *tal que* $P_V(w) = t$. *Pode-se concluir então que* $\exists w \in K : P_V(w) = t$, *onde* $\{w\}$ *é vu-controlável e.r.a.* L. *Sendo assim,* $w \in Sup\mathbb{C}_{VU}(K)$, *o que implica* $P_V(w) = t \in P_V[Sup\mathbb{C}_{VU}(K)]$. *Logo,* $Sup\mathbb{C}_V(E) \subseteq P_V[Sup\mathbb{C}_{VU}(K)]$.

Por fim, resta mostrar que $Sup\mathbb{C}_V(E) \neq \emptyset$ *se e somente se* $Sup\mathbb{C}_{VU}(K) \neq \emptyset$.

(Se) *Seja* $K = P_V^{-1}(E) \cap L$, *onde* $E \subseteq P_V(L)$. *Supor inicialmente que* $\epsilon \in Sup\mathbb{C}_{VU}(K)$ *e que portanto* $Sup\mathbb{C}_{VU}(K) \neq \emptyset$. *Como* $P_V(\epsilon) = \epsilon$ *e como* $Sup\mathbb{C}_V(E) = P_V[Sup\mathbb{C}_{VU}(K)]$, *então* $\epsilon \in Sup\mathbb{C}_{VU}(K) \Rightarrow \epsilon \in Sup\mathbb{C}_V(E)$. *Portanto* $Sup\mathbb{C}_V(E) \neq \emptyset$. *Supor agora que* $\epsilon \notin Sup\mathbb{C}_{VU}(K)$, *mas que* $Sup\mathbb{C}_{VU}(K) \neq \emptyset$. *Pela definição feita para a projeção* $P_V : (V^+ \times U)^* \rightarrow V^*$ *tem-se que* $P_V(\epsilon) = \epsilon$ *e* $P_V(v_0 u_i) = \epsilon$ *para todo* $u_i \in U$. *Desta forma, se* $\epsilon \notin Sup\mathbb{C}_{VU}(K)$ *então* $\epsilon \notin Sup\mathbb{C}_V(E)$. *Mas se* $\epsilon \notin Sup\mathbb{C}_V(E)$ *e* $P_V(v_0 u_i) = \epsilon$, *então* $v_0 u_i \notin Sup\mathbb{C}_{VU}(K)$ *para nenhum* $u_i \in U$. *Supondo finalmente que* $v_0 u_i \circ v_x u_y \circ \ldots \in Sup\mathbb{C}_{VU}(K)$ *para algum* i, x *e* y, *tem-se que* $v_x \circ \ldots \in Sup\mathbb{C}_V(E)$ *e que portanto* $Sup\mathbb{C}_V(E) \neq \emptyset$. *Sendo assim,* $Sup\mathbb{C}_{VU}(K) \neq \emptyset \Longrightarrow Sup\mathbb{C}_V(E) \neq \emptyset$.

(Somente se) *Supor que* $Sup\mathbb{C}_{VU}(K) = \emptyset$. *Sabe-se que* $Sup\mathbb{C}_V(E) = P_V[Sup\mathbb{C}_{VU}(K)]$ *e também que* $P_V(\emptyset) = \emptyset$. *Logo,* $Sup\mathbb{C}_{VU}(K) = \emptyset$ *implica* $Sup\mathbb{C}_V(E) = \emptyset$. *Sendo assim, pode-se afirmar que para que* $Sup\mathbb{C}_V(E) \neq \emptyset$ *é necessário que* $Sup\mathbb{C}_{VU}(K) \neq \emptyset$.

$\diamond$

Desta forma, a máxima linguagem contida em E que é v-controlável em relação a L, é igual à projeção em V^* da máxima linguagem contida em K que é vu-controlável em relação a L. Note que se E consiste na especificação sobre o comportamento lógico do sistema sob supervisão, feita em termos seqüências de eventos, então K consiste na projeção de E em $(V^+ \times U)^*$ e é obtida habilitando-se todas as condições possíveis de ocorrer na planta para cada cadeia de eventos em E.

Pode-se enunciar agora o seguinte corolário.

Corolário 3.1 *Seja uma planta $H = (L, \Gamma)$ e uma especificação $E \subseteq P_V(L)$. Existe um SMN F para H tal que $P_V[L_m(F/H)] = Sup\mathbb{C}_V(E) \neq \emptyset$ se e somente se $Sup\mathbb{C}_{VU}(P_V^{-1}(E) \cap L) \neq \emptyset$.*

Prova: *Segue diretamente das proposições 3.5 e 3.7.* ◇

Retomando o problema de controle supervisório equivalente (PCSE) para sistemas híbridos, o teorema enunciado a seguir introduz uma condição necessária e suficiente para que o PCSE tenha solução.

Teorema 3.1 *O Problema de Controle Supervisório Equivalente para Sistemas Híbridos (PCSE) possui solução se e somente se $Sup\mathbb{C}_{VU}(P_V^{-1}(E) \cap L) \neq \emptyset$.*

Prova:

(Se) *Supor que $Sup\mathbb{C}_{VU}(P_V^{-1}(E) \cap L) \neq \emptyset$. De acordo com o Corolário 3.1, pode-se afirmar que existe um SMN F para H tal que $P_V[L_m(F/H)] = Sup\mathbb{C}_V(E) \neq \emptyset$. Agora, como $Sup\mathbb{C}_V(E) \subseteq E$, pode-se concluir que existe um SMN F para H tal que $\emptyset \neq P_V[L_m(F/H)] \subseteq E$, ou seja, o PCSE possui solução.*

(Somente se) *Supor agora que $Sup\mathbb{C}_{VU}(P_V^{-1}(E) \cap L) = \emptyset$. Ainda pelo Corolário 3.1, sabe-se que não existe um SMN F para H tal que $P_V[L_m(F/H)] \neq \emptyset$, ou seja, o PCSE não possui solução.*

◇

Portanto, um supervisor F para H tal que $L_m(F/H) = Sup\mathbb{C}_{VU}(P_V^{-1}(E) \cap L) \neq \emptyset$ é uma solução ótima para o PCSE no sentido de ser a menos restritiva possível.

Os resultados apresentados anteriormente indicam que a solução do PCSE requer a execução dos seguintes passos:

1. Dada a especificação $E \subseteq V^*$, obter a especificação equivalente $K \subseteq L \subseteq (V^+ \times U)^*$ tal que $P_V(K) = E$;

2. Verificar se K é vu-controlável em relação a L. Se for, pular para o passo 3. Caso contrário, deve-se obter a máxima linguagem vu-controlável contida em K, denotada por $Sup\mathbb{C}_{VU}(K)$;

3. Se $Sup\mathbb{C}_{VU}(K) \neq \emptyset$, então o supervisor F é implementado por intermédio de um autômato tal que $L_m(F/H) = Sup\mathbb{C}_{VU}(K)$.

   Pode-se determinar ainda a máxima linguagem v-controlável contida em E, fazendo $Sup\mathbb{C}_V(E) = P_V[Sup\mathbb{C}_{VU}(K)]$.

A execução dos passos supracitados pode ser feita com auxílio de uma ferramenta chamada GRAIL (Raymond e Wood, 1995), a qual consiste em uma biblioteca de funções em C++ para manejo de autômatos e expressões regulares. Tal biblioteca foi ampliada em (González, 2000) de forma a incluir funções de controle supervisório e também em (da Cunha, 2003) para incluir funções de controle hierárquico com marcação flexível (Cury et al., 2001; Cury et al., 2004; da Cunha e Cury, 2002; Torrico, 2003). Além disso, em (Rodrigues, 2004) foram incluídas funções específicas para o controle supervisório modular de sistemas híbridos.

A seguir, ilustra-se a metodologia de resolução de PCSEs por intermédio de dois exemplos.

Exemplo 3.7 *Seja um sistema híbrido $\mathcal{H}_1$ cujo comportamento lógico $L_1 = \mathcal{L}(\mathcal{H}_1)$ é reconhecido pelo autômato mostrado na Figura 3.14.*

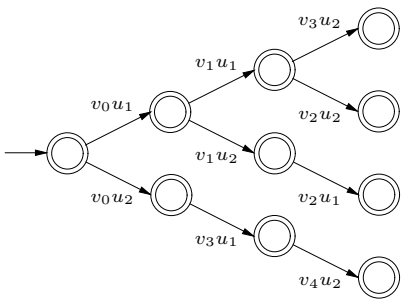
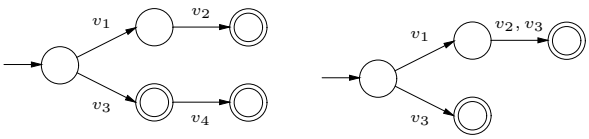


Figura 3.14: Autômato que reconhece a linguagem $L_1 = \mathcal{L}(\mathcal{H}_1)$.

Sejam ainda duas especificações $E_1, E_2 \subseteq P_V(L_1)$ mostradas na Figura 3.15.



(a) $E_1 \subseteq P_V(L_1)$ (b) $E_2 \subseteq P_V(L_1)$

Figura 3.15: Especificações $E_1, E_2 \subseteq P_V(L_1)$.

O primeiro passo para a síntese do supervisor monolítico consiste em encontrar uma especificação única $E \subseteq V^*$, a qual é ilustrada pela Figura 3.16. Esta especificação global é obtida fazendo-se a intersecção das especificações individuais.

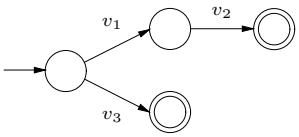


Figura 3.16: Especificação global $E \subseteq V^*$.

A seguir, obtém-se uma especificação equivalente que esteja contida na linguagem da planta, ou seja, $K \subseteq L_1 \subseteq (V^+ \times U)^*$. Tal especificação é obtida fazendo-se $K = P_V^{-1}(E) \cap L_1$ e o autômato que a reconhece é mostrado na Figura 3.17.

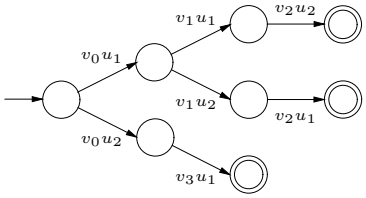


Figura 3.17: Especificação $K \subseteq L_1$.

Verifica-se, entretanto, que a linguagem K não é vu-controlável em relação a L_1. Note, por exemplo, que após a seqüência $w = v_0u_2 \circ v_3u_1 \in \overline{K}$ tenta-se desabilitar o evento v_4 previsto na planta. Da mesma forma, após a cadeia $w' = v_0u_1 \circ v_1u_1 \in \overline{K}$, a especificação não prevê a ocorrência do evento v_3, o qual pode ocorrer na planta. Obtém-se então a máxima linguagem vu-controlável contida em K, a qual é reconhecida pelo autômato mostrado na Figura 3.18.

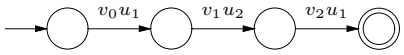


Figura 3.18: Máxima linguagem vu-controlável $Sup\mathbb{C}_{VU}(K)$.

A máxima linguagem v-controlável contida na especificação $E \subseteq V^*$ pode ser obtida fazendo-se $Sup\mathbb{C}_V(E) = P_V[Sup\mathbb{C}_{VU}(K)]$. Neste exemplo tem-se que $Sup\mathbb{C}_V(E) = v_1 \circ v_2$.

Por fim, um supervisor F_1 tal que $L_m(F_1/H_1) = Sup\mathbb{C}_{VU}(K)$ consiste numa solução ótima para o problema apresentado neste exemplo. Este supervisor pode ser implementado pelo autômato mostrado na Figura 3.18.

■

Exemplo 3.8 *Seja um sistema híbrido $\mathcal{H}_2$ com $V = \{v_1, v_2, v_3, v_4, v_5\}$ e $U = \{u_1, u_2\}$, baseado em Koutsoukos et al. (2000). O comportamento lógico $L_2 = \mathcal{L}(\mathcal{H}_2)$ deste sistema é reconhecido pelo autômato da Figura 3.19.*

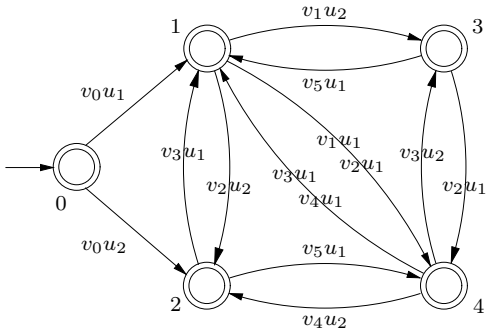


Figura 3.19: Autômato que reconhece a linguagem $L_2 = \mathcal{L}(\mathcal{H}_2)$.

Considere uma especificação $E' \subseteq V^*$ na qual deseja-se que seja garantida a ocorrência do evento v_5 uma única vez, conforme ilustrado pela Figura 3.20.

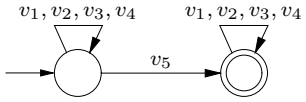


Figura 3.20: Especificação $E' \subseteq V^*$.

Na Figura 3.21 representa-se a linguagem-alvo $K' = P_V^{-1}(E') \cap L_2$.

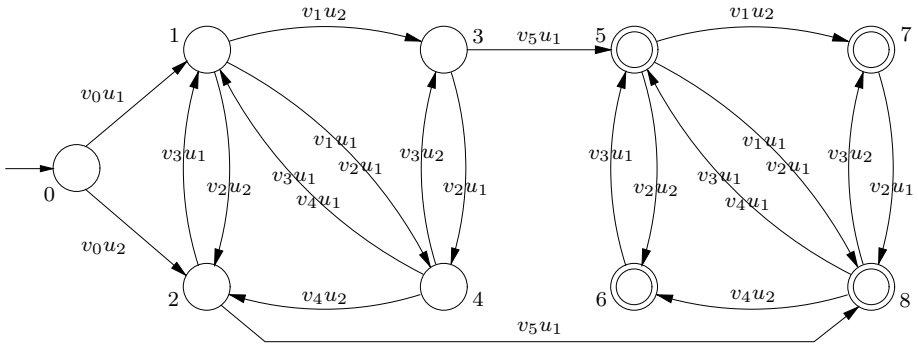


Figura 3.21: Linguagem-alvo $K' \subseteq L_2$.

Pode-se verificar que a linguagem-alvo não é vu-controlável em relação a L_2 *uma vez que nos estados* 6 *e* 7 *do autômato da Figura 3.21 não é prevista a ocorrência do evento* v_5*, sendo que o mesmo pode ocorrer na planta (estados* 2 *e* 3 *da planta). A máxima linguagem vu-controlável* $Sup\mathbb{C}_{VU}(K')$ *é reconhecida pelo autômato mostrado na Figura 3.22.*

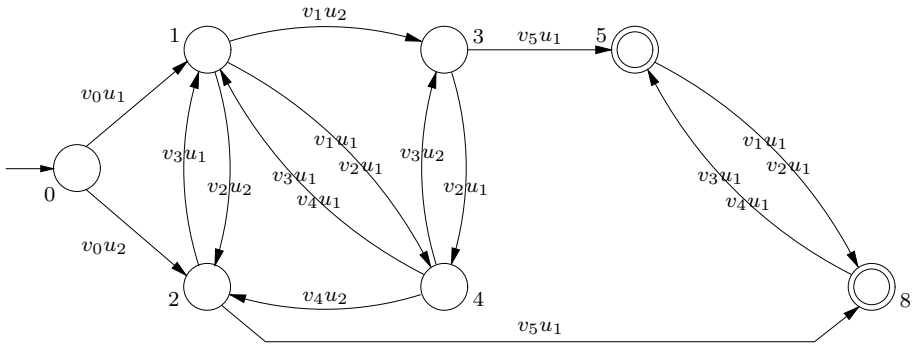


Figura 3.22: Máxima linguagem vu-controlável $Sup\mathbb{C}_{VU}(K')$.

A máxima linguagem v-controlável contida na especificação $E' \subseteq V^*$ *pode ser obtida fazendo-se* $Sup\mathbb{C}_V(E') = P_V[Sup\mathbb{C}_{VU}(K')]$. *Esta linguagem não é mostrada, mas pode-se facilmente observar por intermédio do autômato para* $Sup\mathbb{C}_{VU}(K')$ *que a especificação* E' *está garantida.*

Por fim, um supervisor F_2 *tal que* $L_m(F_2/H_2) = Sup\mathbb{C}_{VU}(K')$ *consiste numa solução ótima para este exemplo. Este supervisor pode ser implementado pelo autômato mostrado na Figura 3.22.*

■

O Corolário 3.2 apresenta uma condição para que o Problema de Síntese de Supervisores para Sistemas Híbridos (SSSH) tenha solução, e ainda indica como obter uma solução para o mesmo através da resolução do PCSE.

Corolário 3.2 *O Problema de Síntese de Supervisores para Sistemas Híbridos (SSSH) possui solução se e somente se o Problema de Controle Supervisório Equivalente para Sistemas Híbridos (PCSE) possui solução. Além disso, dado um SMN F como uma solução para o PCSE, o supervisor C/E $\mathcal{F}$ obtido por intermédio do Algoritmo 1 apresentado em (González et al., 2001) é a solução correspondente para o SSSH.*

Prova: *Segue diretamente da Proposição 3.1.* ◇

Conforme estabelecido no Corolário 3.2, a resolução do Problema de Síntese de Supervisores para Sistemas Híbridos (SSSH) pode ser feita por intermédio da resolução de um problema equivalente no domínio de SEDs, o chamado PCSE.

A resolução de um problema de síntese de supervisores para sistemas híbridos requer a execução dos seguintes passos:

1. Obter um modelo que represente o comportamento lógico do sistema híbrido a ser controlado;

2. Formular e resolver o problema de controle supervisório (PCSE) equivalente ao SSSH, utilizando para tal o modelo obtido no passo anterior;

3. O supervisor SED obtido na resolução do PCSE é utilizado como uma implementação discreta do supervisor C/E, não sendo necessária a determinação deste último.

A resolução de um exemplo de síntese de supervisores para sistemas híbridos (SSSH) foi deixada para o Capítulo 4, no qual o problema será formulado e solucionado mostrando a sua resolução desde a extração do modelo lógico até a obtenção dos supervisores. A resolução de um problema prático, no qual é feita a implementação física do controle supervisório, pode ser encontrada em (de Paula e Silva, 2004).

3.4 Conclusões

Neste capítulo, trata-se do controle supervisório de uma classe de sistemas híbridos que possui dinâmicas contínuas e discretas interagindo entre si. Esta classe de sistemas híbridos consiste em uma generalização daquela abordada nos trabalhos de Cury et al. (1998), Koutsoukos et al. (2000) e Moor et al. (1998), nos quais o caráter discreto advém somente do supervisor, ou seja, o aspecto híbrido daqueles trabalhos consiste na reunião de uma planta contínua com um supervisor discreto. O problema de controle supervisório para sistemas híbridos (SSSH) é formulado por intermédio do paradigma de modelagem de sistemas condição/evento e mostra-se que este problema pode ser solucionado por intermédio da resolução de um problema equivalente no domínio discreto, o qual é denominado Problema de Controle Supervisório Equivalente (PCSE).

Dois exemplos são utilizados de forma a ilustrar a resolução do problema de controle supervisório equivalente. Em ambos os exemplos consideram-se conhecidos os modelos lógicos para os respectivos sistemas.

A principal contribuição deste capítulo consiste na extensão da teoria de controle supervisório para sistemas híbridos proposta por González et al. (2001). Além disso, definições importantes foram introduzidas neste capítulo, tais como as de supervisor consistente, supervisor marcador não bloqueante, vu-controlabilidade e v-controlabilidade, sendo estas também contribuições deste trabalho. De uma forma geral, pode-se dizer que este trabalho contribui para a consolidação da teoria de controle supervisório para sistemas híbridos.

Capítulo 4

Controle Modular de Sistemas Híbridos

Neste capítulo trata-se da abordagem modular de síntese de supervisores para a classe de sistemas híbridos apresentada no Capítulo 3. É neste capítulo que se concentram as maiores contribuições do presente trabalho, sendo que boa parte delas foi previamente apresentada em três artigos: em (Leal e Cury, 2002) trata-se do controle modular de sistemas condição/evento; em (Leal e Cury, 2004a) aborda-se o problema de síntese de supervisores modulares para sistemas híbridos; e em (Leal e Cury, 2004b) estendem-se os resultados obtidos anteriormente de forma a considerar marcação nas linguagens.

4.1 Introdução

Freqüentemente, o comportamento desejado para a planta sob supervisão é expresso pela combinação de duas ou mais especificações (restrições). Em outras palavras, uma especificação para o comportamento desejado é comumente descrita na forma: "O sistema deve atender uma propriedade de um tipo *bem como* uma propriedade de outro tipo".

Por abordagem de controle modular, entende-se aquela em que se projeta um controlador individual para atender cada um dos componentes da especificação global, e os supervisores modulares são implementados de forma concorrente no intuito de que a ação conjunta destes garanta o cumprimento da especificação global. O projeto individual dos supervisores modulares é feito conforme tratado no Capítulo 3.

A abordagem de controle modular, de uma forma geral, possui duas grandes motivações:

diminuir a complexidade computacional em relação à abordagem monolítica; e aumentar a flexibilidade em caso de mudanças: se uma especificação relativa a uma subtarefa é alterada, por exemplo, pode-se reprojetar apenas o supervisor referente àquela especificação. Por outro lado, como a ação de controle de cada supervisor modular não considera as ações dos demais supervisores, pode ser que dois ou mais supervisores entrem em conflito ao atuarem em conjunto para atingir uma especificação global (Wonham e Ramadge, 1988). Sendo assim, a síntese de supervisores modulares deve ser feita de forma a garantir que a ação conjunta dos supervisores obtidos seja não bloqueante e minimamente restritiva em relação à especificação global.

A síntese modular de supervisores para sistemas híbridos foi abordada anteriormente apenas em (Moor, Davoren e Raisch, 2001), sendo que naquele trabalho o problema é formulado usando a teoria comportamental de sistemas (Willems, 1991), enquanto que aqui a formulação é feita por intermédio da teoria de sistemas condição/evento (Sreenivas e Krogh, 1991). Existem várias outras diferenças entre estes trabalhos, mas as principais são brevemente discutidas na seção que encerra este capítulo.

O restante do capítulo está estruturado como segue. Na próxima seção, o problema de síntese modular de supervisores para a classe de sistemas híbridos apresentada no Capítulo 3 é formulado utilizando-se o formalismo de sistemas condição/evento (Sreenivas e Krogh, 1991). Na Seção 4.3 mostra-se que o problema anteriormente formulado pode ser solucionado utilizando-se uma abordagem de controle no domínio discreto. Na Seção 4.4 apresenta-se um exemplo de forma a ilustrar a metodologia proposta para a síntese modular de supervisores para sistemas híbridos. Finalmente, na Seção 4.5 comenta-se sobre a abordagem proposta e faz-se uma discussão sobre os problemas que serão tratados no seguimento deste trabalho.

4.2 Formulação do Problema

Considere o esquema de controle supervisório modular da Figura 4.1 para a classe de sistemas híbridos apresentada no Capítulo 3. Esta classe de sistemas híbridos é tal que a seleção da dinâmica contínua é dirigida por eventos de limiar, os quais são gerados sempre que variáveis de estado contínuo alcançam uma superfície de limiar. Deve-se destacar que por motivos de simplicidade, considera-se o caso em que são utilizados apenas dois supervisores.

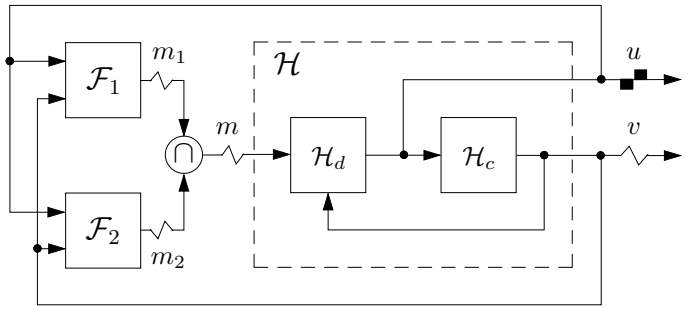


Figura 4.1: Esquema de controle modular para a planta híbrida.

Definição 4.1 *Sejam $m_1(\cdot), m_2(\cdot) \in \mathcal{M}$ dois sinais evento síncronos tomando valores em $M = 2^U$. Define-se a intersecção síncrona $m_1(\cdot) \bigcirc\!\!\!\!\cap\, m_2(\cdot)$ destes sinais como o sinal evento $m(\cdot)$ tal que $m(t) = m_1(t) \cap m_2(t), t \in [0, \infty)$.*

A Figura 4.2 mostrada a seguir ilustra este conceito, sendo que $m(\cdot) = m_1(\cdot) \bigcirc\!\!\!\!\cap\, m_2(\cdot)$.

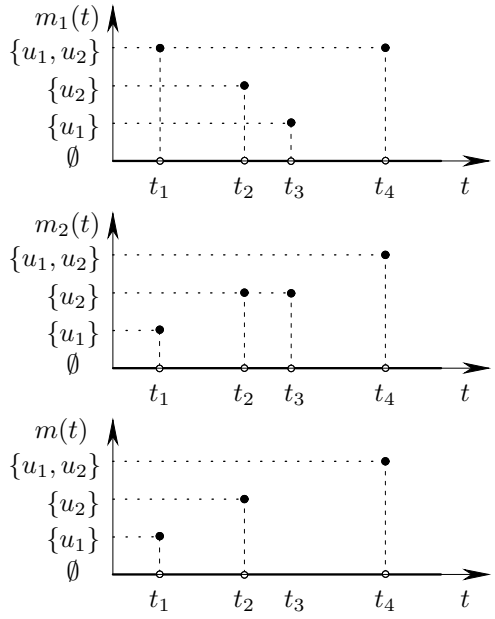


Figura 4.2: Exemplo de intersecção síncrona de dois sinais evento.

Observe, por exemplo, que no instante t_1 o sinal $m(\cdot)$ assume o valor $m(t_1) = m_1(t_1) \bigcirc\!\!\!\!\cap\, m_2(t_1) = \{u_1, u_2\} \cap \{u_1\} = \{u_1\}$.

Definição 4.2 *Sejam $\mathcal{F}_1$ e $\mathcal{F}_2$ supervisores C/E consistentes para uma planta híbrida $\mathcal{H}$. A conjunção de $\mathcal{F}_1$ e $\mathcal{F}_2$ é definida como um sistema C/E $\mathcal{F}_1 \wedge \mathcal{F}_2$ tal que:*

$$(v(\cdot), m(\cdot), u(\cdot)) \in (\mathcal{F}_1 \wedge \mathcal{F}_2) \Longleftrightarrow m(\cdot) = m_1(\cdot) \textcircled{\cap} m_2(\cdot)$$

onde $(v(\cdot), m_i(\cdot), u(\cdot)) \in \mathcal{F}_i, \quad i = 1, 2.$

A seguir, apresenta-se o problema de síntese de supervisores modulares para sistemas híbridos – SMSH.

Problema 4.1 (Síntese Modular para SHs – SMSH) *Seja $\mathcal{H}$ o modelo C/E da planta híbrida e sejam $E_1, E_2 \subseteq V^*$ especificações para o comportamento da planta em malha fechada. Encontrar SCEMN $\mathcal{F}_i$ $(i = 1, 2)$ tais que a conjunção $\mathcal{F}_1 \wedge \mathcal{F}_2$ seja um SCEMN em relação a $E_1 \cap E_2$ e que $\emptyset \neq P_V[\mathcal{L}_m((\mathcal{F}_1 \wedge \mathcal{F}_2)/\mathcal{H})] \subseteq E_1 \cap E_2$.*

O problema de síntese de supervisores modulares para sistemas híbridos consiste então em projetar supervisores $\mathcal{F}_i$, cada um para satisfazer uma especificação própria, e implementar o controle global pela conjunção dos supervisores $\mathcal{F}_i$ projetados. Deve-se observar que o projeto dos supervisores individuais é feito conforme apresentado no Capítulo 3.

De forma análoga ao que foi feito no Capítulo 3, na próxima seção utiliza-se a teoria de controle supervisório de SEDs de modo a propor uma solução formal para o Problema SMSH.

4.3 Abordagem por Controle de Sistemas a Eventos Discretos

Aqui mostra-se que, diferentemente do que ocorre na teoria clássica de controle modular de SEDs (Wonham e Ramadge, 1988), para os sistemas condição/evento não basta que as linguagens sejam não conflitantes para que se possa utilizar a abordagem modular. Introduz-se então o conceito de *interconsistência* entre linguagens (Leal e Cury, 2002). Vale salientar que nesta seção apresenta-se uma série de lemas de forma a auxiliar as demonstrações de resultados enunciados nas proposições e também no Teorema 4.1, o qual consiste no principal resultado da seção. Assim, caso o leitor deseje fazer uma leitura mais rápida, recomenda-se a análise das definições de *conjunção de supervisores* (Definição 4.3) e de *interconsistência entre linguagens* (Definição 4.4), seguida da leitura do Teorema 4.1. Depois, de acordo com a necessidade, o leitor pode retornar e fazer uma consulta àqueles ítens que se fizerem necessários.

A seguir, define-se a conjunção de supervisores SED.

Definição 4.3 *Sejam F_1 e F_2 supervisores consistentes para uma planta $H = (L, \Gamma)$. A conjunção de F_1 e F_2 é definida como um supervisor $F_1 \wedge F_2$ dado pelo mapa $(F_1 \wedge F_2) : L \rightarrow 2^{V^+ \times U}$ tal que para $w \in L$:*

$$vu \in (F_1 \wedge F_2)(w) \Longleftrightarrow vu \in (F_1(w) \cap F_2(w)).$$

Sendo assim, um par $vu \in V^+ \times U$ é habilitado por $F_1 \wedge F_2$ se e somente se vu é habilitado simultaneamente por F_1 e F_2.

Antes de analisar em detalhe a ação de $F_1 \wedge F_2$, introduz-se o conceito de *interconsistência* entre linguagens e apresentam-se alguns resultados intermediários.

Definição 4.4 *Duas linguagens $K_1, K_2 \subseteq (V^+ \times U)^*$ são ditas interconsistentes se:*

$$(\forall w \in \overline{K_1} \cap \overline{K_2})\ V_{K_1}(w) \cap V_{K_2}(w) = V_{K_1 \cap K_2}(w). \tag{4.1}$$

Como $V_{K_1}(w) \cap V_{K_2}(w) \supset V_{K_1 \cap K_2}(w)$ sempre é verdade, então $V_{K_1}(w) \cap V_{K_2}(w) \neq V_{K_1 \cap K_2}(w)$ implica $V_{K_1}(w) \cap V_{K_2}(w) \nsubseteq V_{K_1 \cap K_2}(w)$. Assim, duas linguagens, K_1 e K_2, são *interconsistentes* se, após qualquer cadeia comum a $\overline{K_1}$ e $\overline{K_2}$, os eventos possíveis de ocorrer em ambas as linguagens são também possíveis de ocorrer na linguagem $K_1 \cap K_2$. A equação (4.1) usada para o teste da interconsistência pode ser substituída pela equação mostrada abaixo.

$$(\forall w \in \overline{K_1} \cap \overline{K_2})\ (\forall v \in V_{K_1}(w) \cap V_{K_2}(w))\ (U_{K_1}(w, v) \cap U_{K_2}(w, v) \neq \emptyset). \tag{4.2}$$

No intuito de ilustrar o conceito de interconsistência, apresenta-se a seguir um exemplo de linguagens não interconsistentes.

Exemplo 4.1 *Sejam as linguagens $K_1 = v_1u_1 + v_1u_1 \circ v_2u_1 \circ v_3u_1$ e $K_2 = v_1u_1 + v_1u_1 \circ v_2u_2 \circ v_4u_2$. Assim, $\overline{K_1} \cap \overline{K_2} = \{\epsilon, v_1u_1\}$. Note que para $w = v_1u_1 \in \overline{K_1} \cap \overline{K_2}$ tem-se que $V_{K_1}(w) = V_{K_2}(w) = \{v_2\}$, mas que $V_{K_1 \cap K_2}(w) = \emptyset$, ou seja, $\exists w \in \overline{K_1} \cap \overline{K_2}$ tal que $V_{K_1}(w) \cap V_{K_2}(w) \nsubseteq V_{K_1 \cap K_2}(w)$. Sendo assim, conclui-se que K_1 e K_2 não são interconsistentes. A mesma conclusão é obtida utilizando-se a equação (4.2) para realizar o teste da interconsistência. Para o mesmo $w = v_1u_1 \in \overline{K_1} \cap \overline{K_2}$ e para $v_2 \in V_{K_1}(w) \cap V_{K_2}(w)$ tem-se que $U_{K_1}(w, v_2) = \{u_1\}$ e $U_{K_2}(w, v_2) = \{u_2\}$. Logo, pode-se afirmar que $(\exists w \in \overline{K_1} \cap \overline{K_2})\ (\exists v \in V_{K_1}(w) \cap V_{K_2}(w)) : (U_{K_1}(w, v) \cap U_{K_2}(w, v) = \emptyset)$, ou seja, que K_1 e K_2 não são interconsistentes.* ■

Na abordagem de Ramadge e Wonham para o controle supervisório modular de SEDs, duas linguagens K_1, K_2 são ditas modulares (ou não conflitantes) se $\overline{K1} \cap \overline{K2} = \overline{K1 \cap K2}$. Na teoria de controle supervisório de SEDs, a modularidade entre linguagens é uma condição necessária e suficiente para que se possa garantir a otimalidade da solução encontrada através da abordagem modular (Wonham e Ramadge, 1988). A seguir, mostra-se que na abordagem proposta no presente trabalho a modularidade não é suficiente e mostra-se ainda que a interconsistência é uma condição necessária e suficiente para que se possa utilizar a abordagem modular para o controle supervisório de sistemas híbridos.

Proposição 4.1 *Sejam K_1 e K_2 duas linguagens vu-controláveis e.r.a. (em relação a) linguagem L. Se K_1 e K_2 são interconsistentes, então K_1 e K_2 são modulares.*

Prova: *Prova-se que se K_1 e K_2 não são modulares, então K_1 e K_2 não são interconsistentes.*

Sejam K_1 e K_2 linguagens vu-controláveis e.r.a. L. Supor que K_1 e K_2 não são modulares, então $\overline{K_1} \cap \overline{K_2} \neq \overline{K_1 \cap K_2}$. Sabe-se que $\overline{K_1} \cap \overline{K_2} \supseteq \overline{K_1 \cap K_2}$ sempre é verificado, logo a não modularidade implica $\overline{K_1} \cap \overline{K_2} \nsubseteq \overline{K_1 \cap K_2}$. Ou seja, $\exists w \in \overline{K_1} \cap \overline{K_2}$ tal que $w \notin \overline{K_1 \cap K_2}$. Seja então uma cadeia $w \in \overline{K_1} \cap \overline{K_2}$ tal que $w \notin \overline{K_1 \cap K_2}$. Sabe-se que para esta cadeia existem $s, s' \in (V \times U)^*$ tais que $w \circ s \in K_1$ e $w \circ s' \in K_2$, onde $s \neq s'$. Supondo que $s' = \epsilon$, como $s \neq s'$ tem-se que $s \neq \epsilon$. Desta forma, pode-se afirmar que $w \circ s \in K_1$, sendo $s \neq \epsilon$, de onde se conclui que $V_{K_1}(w) \neq \emptyset$. Agora, como K_1 e K_2 são vu-controláveis e.r.a. L, para $w \in \overline{K_1} \cap \overline{K_2}$ tem-se que $V_{K_1}(w) = V_L(w)$ e $V_{K_2}(w) = V_L(w)$, ou seja, $V_{K_1}(w) = V_{K_2}(w) = V_L(w) \neq \emptyset$. Desta forma, tem-se que $V_{K_1}(w) \cap V_{K_2}(w) \neq \emptyset$. Por fim, como $w \notin \overline{K_1 \cap K_2}$, tem-se que $V_{K_1 \cap K_2}(w) = \emptyset$, de onde se pode concluir que existe $w \in \overline{K_1} \cap \overline{K_2}$ tal que $V_{K_1}(w) \cap V_{K_2}(w) \nsubseteq V_{K_1 \cap K_2}(w)$, ou seja, que K_1 e K_2 não são interconsistentes. ◇

Observe que muito embora a interconsistência entre linguagens *vu*-controláveis implique a modularidade entre as mesmas, o inverso não é verdade. Ou seja, a modularidade entre linguagens *vu*-controláveis não implica, necessariamente, a interconsistência entre estas. Esta propriedade é ilustrada pelo exemplo apresentado a seguir.

Exemplo 4.2 *Seja o autômato mostrado na Figura 4.3, o qual reconhece a linguagem* $L = \overline{v_1u_1 \circ v_2u_2 \circ (v_3u_1 \circ v_4u_2 + v_3u_2 \circ v_5u_1)}$.

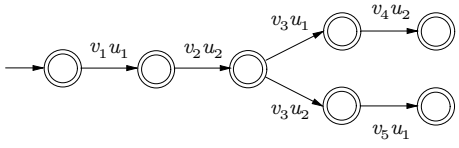
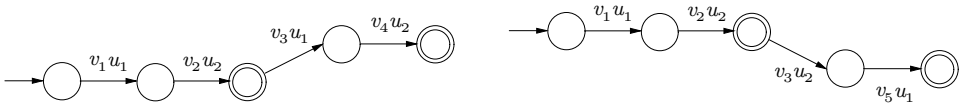


Figura 4.3: Autômato que reconhece a linguagem L do Exemplo 4.2.

Sejam ainda K_1 *e* K_2 *duas linguagens vu-controláveis e.r.a. L. Na Figura 4.4 apresentam-se dois autômatos que reconhecem estas linguagens.*



(a) Autômato que reconhece K_1. (b) Autômato que reconhece K_2.

Figura 4.4: Linguagens vu-controláveis em relação à L.

Pode ser facilmente verificado que $\overline{K_1} \cap \overline{K_2} = \overline{K_1 \cap K_2}$ *(ver Figura 4.5), de onde se conclui que as linguagens* K_1 *e* K_2 *são modulares.*

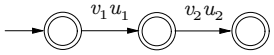


Figura 4.5: Autômato que reconhece a linguagem $\overline{K_1} \cap \overline{K_2} = \overline{K_1 \cap K_2}$.

Observe agora que para $w = v_1u_1 \circ v_2u_2 \in \overline{K_1} \cap \overline{K_2}$ *tem-se que* $V_{K_1}(w) \cap V_{K_2}(w) = \{v_3\}$, *mas que* $V_{K_1 \cap K_2}(w) = \emptyset$, *ou seja,* $\exists w \in \overline{K_1} \cap \overline{K_2} : V_{K_1}(w) \cap V_{K_2}(w) \not\subseteq V_{K_1 \cap K_2}(w)$. *Sendo assim, embora as linguagens* K_1 *e* K_2 *sejam modulares, elas não são interconsistentes.* ■

Proposição 4.2 *Sejam* $K_1, K_2 \subseteq (V^+ \times U)^*$ *linguagens vu-controláveis e.r.a. L. Se* K_1 *e* K_2 *são interconsistentes, então a linguagem* $K_1 \cap K_2$ *também é vu-controlável e.r.a. L.*

Prova: *Sejam* K_1 *e* K_2 *linguagens vu-controláveis e.r.a. L, então* $(\forall w \in \overline{K_1})\ V_{K_1}(w) = V_L(w)$ *e* $(\forall w \in \overline{K_2})\ V_{K_2}(w) = V_L(w)$. *Assim, para* $w \in \overline{K_1} \cap \overline{K_2}$ *tem-se que* $V_{K_1}(w) \cap V_{K_2}(w) = V_L(w)$. *Supor agora que* K_1 *e* K_2 *são interconsistentes, então* $(\forall w \in \overline{K_1} \cap \overline{K_2})\ V_{K_1}(w) \cap V_{K_2}(w) = V_{K_1 \cap K_2}(w)$, *de onde se conclui que*

$$(\forall w \in \overline{K_1} \cap \overline{K_2})\ V_{K_1 \cap K_2}(w) = V_L(w). \tag{4.3}$$

Pela Proposição 4.1 sabe-se que se K_1 e K_2 são interconsistentes, então K_1 e K_2 são modulares. Assim, $\overline{K_1} \cap \overline{K_2} = \overline{K_1 \cap K_2}$ e a equação (4.3) pode ser escrita como

$$(\forall w \in \overline{K_1 \cap K_2})\ V_{K_1 \cap K_2}(w) = V_L(w).$$

Logo, $K_1 \cap K_2$ é vu-controlável em relação a L. $\diamond$

A seguir, apresenta-se um resultado parcial que será utilizado para auxiliar a demonstração de resultados subseqüentes.

Lema 4.1 $P_V^{-1}(E_1) \cap P_V^{-1}(E_2) = P_V^{-1}(E_1 \cap E_2)$.

Prova: *Sabe-se que $P_V^{-1}(E_1) = \{w_1 \in (V^+ \times U)^* : P_V(w_1) \in E_1\}$ e que $P_V^{-1}(E_2) = \{w_2 \in (V^+ \times U)^* : P_V(w_2) \in E_2\}$. Assim, $P_V^{-1}(E_1) \cap P_V^{-1}(E_2) = \{w \in (V^+ \times U)^* : P_V(w) \in E_1 \cap E_2\} = P_V^{-1}(E_1 \cap E_2)$.* $\diamond$

Proposição 4.3 *Seja F_i um SMN em relação a E_i, com $i = 1, 2$. Então $L_m((F_1 \wedge F_2)/H) = L_m(F_1/H) \cap L_m(F_2/H)$, onde $L_m((F_1 \wedge F_2)/H) = L((F_1 \wedge F_2)/H) \cap P_V^{-1}(E_1 \cap E_2)$. Além disso, $F_1 \wedge F_2$ é um supervisor consistente para H se e somente se as linguagens $L_m(F_1/H)$ e $L_m(F_2/H)$ são interconsistentes.*

Prova: *Inicialmente prova-se que $L((F_1 \wedge F_2)/H) = L(F_1/H) \cap L(F_2/H)$.*

- $L((F_1 \wedge F_2)/H) \supseteq L(F_1/H) \cap L(F_2/H)$

  Seja $w \circ vu \in (L(F_1/H) \cap L(F_2/H))$, então $w \in (L(F_1/H) \cap L(F_2/H))$ e $vu \in (F_1(w) \cap F_2(w))$. Assume-se que $w \in L((F_1 \wedge F_2)/H)$. Mas se $vu \in (F_1(w) \cap F_2(w))$, então (Definição 4.3) $vu \in (F_1 \wedge F_2)(w)$, e assim $w \circ vu \in L((F_1 \wedge F_2)/H)$. Logo $L((F_1 \wedge F_2)/H) \supseteq L(F_1/H) \cap L(F_2/H)$.

- $L((F_1 \wedge F_2)/H) \subseteq L(F_1/H) \cap L(F_2/H)$

  Seja $w \circ vu \in L((F_1 \wedge F_2)/H)$, então pela definição de $L(F/H)$ pode-se afirmar que $w \in L((F_1 \wedge F_2)/H)$ e $vu \in (F_1 \wedge F_2)(w)$. Assume-se que $w \in (L(F_1/H) \cap L(F_2/H))$. De acordo com a definição de conjunção de supervisores (Definição 4.3) sabe-se que $vu \in (F_1 \wedge F_2)(w)$ implica $vu \in (F_1(w) \cap F_2(w))$ e então $w \circ vu \in (L(F_1/H) \cap L(F_2/H))$. Logo $L((F_1 \wedge F_2)/H) \subseteq L(F_1/H) \cap L(F_2/H)$.

Sendo assim, $L((F_1 \wedge F_2)/H) = L(F_1/H) \cap L(F_2/H)$.

No que segue, mostra-se que $L_m((F_1 \wedge F_2)/H) = L_m(F_1/H) \cap L_m(F_2/H)$.

Sabe-se que $L_m(F_i/H) = L(F_i/H) \cap P_V^{-1}(E_i)$. *Logo:*

$$\begin{aligned} L_m(F_1/H) \cap L_m(F_2/H) &= [L(F_1/H) \cap P_V^{-1}(E_1)] \cap [L(F_2/H) \cap P_V^{-1}(E_2)] \\ &= [L(F_1/H) \cap L(F_2/H)] \cap [P_V^{-1}(E_1) \cap P_V^{-1}(E_2)] \\ &= L((F_1 \wedge F_2)/H) \cap [P_V^{-1}(E_1) \cap P_V^{-1}(E_2)]. \end{aligned}$$

Mas pelo Lema 4.1 tem-se que $P_V^{-1}(E_1) \cap P_V^{-1}(E_2) = P_V^{-1}(E_1 \cap E_2)$ *e portanto:*

$$\begin{aligned} L_m(F_1/H) \cap L_m(F_2/H) &= L((F_1 \wedge F_2)/H) \cap P_V^{-1}(E_1 \cap E_2) \\ &= L_m((F_1 \wedge F_2)/H). \end{aligned}$$

Na segunda parte desta prova, mostra-se que $F_1 \wedge F_2$ *é um supervisor consistente para* H *se e somente se as linguagens* $L_m(F_1/H)$ *e* $L_m(F_2/H)$ *são interconsistentes.*

Sabe-se que um supervisor é consistente para H *se:*

$$(\forall w \in L(F/H))\ F(w) = \gamma \in \Gamma(w). \tag{4.4}$$

Mas $\Gamma(w) = \{\gamma : (\forall v \in V_L(w))(\exists u \in U_L(w,v)) : vu \in \gamma\}$ *e então a equação (4.4) pode ser escrita como:*

$$(\forall w \in L(F/H))\ (\forall v \in V_L(w))\ \exists u \in U_L(w,v) : vu \in F(w).$$

Para simplificar a notação, no que segue, faz-se $L_m(F_i/H) = K_i$.

(Se) *Supor que* K_1 *e* K_2 *sejam interconsistentes, então:*

$$(\forall w \in \overline{K_1} \cap \overline{K_2})\ (\forall v \in V_{K_1}(w) \cap V_{K_2}(w))\ (U_{K_1}(w,v) \cap U_{K_2}(w,v) \neq \emptyset). \tag{4.5}$$

Agora $U_{K_i}(w,v) = \{u \in U : w \circ vu \in \overline{K_i}\}$ *e então* $U_{K_1}(w,v) \cap U_{K_2}(w,v) = \{u \in U : w \circ vu \in \overline{K_1} \cap \overline{K_2}\}$. *Assim, como* $U_{K_1}(w,v) \cap U_{K_2}(w,v) \neq \emptyset$, *tem-se que* $\exists u \in U : w \circ vu \in \overline{K_1} \cap \overline{K_2}$.

Logo, pode-se escrever (4.5) como segue:

$$(\forall w \in \overline{K_1} \cap \overline{K_2})\ (\forall v \in V_{K_1}(w) \cap V_{K_2}(w))\ \exists u \in U : w \circ vu \in \overline{K_1} \cap \overline{K_2}. \tag{4.6}$$

Ainda, como K_1 *e* K_2 *são vu-controláveis e.r.a.* L*, então* $(\forall w \in \overline{K_i})V_{K_i}(w) = V_L(w)$ *e portanto* $(\forall w \in \overline{K_1} \cap \overline{K_2})V_{K_1}(w) \cap V_{K_2}(w) = V_L(w)$*. Desta forma, pode-se escrever (4.6) como:*

$$(\forall w \in \overline{K_1} \cap \overline{K_2})\ (\forall v \in V_L(w))\ \exists u \in U : w \circ vu \in \overline{K_1} \cap \overline{K_2}. \tag{4.7}$$

Mas $\overline{K_1}, \overline{K_2} \subseteq L$ *e então* $w \circ vu \in \overline{K_1} \cap \overline{K_2}$ *implica* $w \circ vu \in L$ *e assim tem-se que* $u \in U_L(w, v)$*. Além disso, uma vez que* F_1 *e* F_2 *são supervisores não bloqueantes para* H *tem-se que* $\overline{L_m(F_i/H)} = \overline{K_i} = L(F_i/H)$ *e assim* $\overline{K_1} \cap \overline{K_2} = L(F_1/H) \cap L(F_2/H) = L((F_1 \wedge F_2)/H)$*. Desta forma, (4.7) pode ser apresentada na seguinte forma:*

$$(\forall w \in L((F_1 \wedge F_2)/H))\ (\forall v \in V_L(w))\ \exists u \in U_L(w, v) : w \circ vu \in L((F_1 \wedge F_2)/H).$$

Por fim, de acordo com a definição de $L(F/H)$ *tem-se que se* $w \circ vu \in L((F_1 \wedge F_2)/H)$ *então* $vu \in (F_1 \wedge F_2)(w)$*, de onde se obtém que:*

$$(\forall w \in L((F_1 \wedge F_2)/H))\ (\forall v \in V_L(w))\ \exists u \in U_L(w, v) : vu \in (F_1 \wedge F_2)(w).$$

Ou seja, $F_1 \wedge F_2$ *é um supervisor consistente para* H*.*

(Somente se) *Supor que* K_1 *e* K_2 *não sejam interconsistentes, então:*

$$(\exists w \in \overline{K_1} \cap \overline{K_2})\ (\exists v \in V_{K_1}(w) \cap V_{K_2}(w)) : U_{K_1}(w, v) \cap U_{K_2}(w, v) = \emptyset\ . \tag{4.8}$$

Mas $U_{K_1}(w, v) \cap U_{K_2}(w, v) = \{u \in U : w \circ vu \in \overline{K_1} \cap \overline{K_2}\}$*. Assim, para* $U_{K_1}(w, v) \cap U_{K_2}(w, v) = \emptyset$*, tem-se que* $\nexists u \in U : w \circ vu \in \overline{K_1} \cap \overline{K_2}$*. Reescreve-se (4.8) como segue:*

$$(\exists w \in \overline{K_1} \cap \overline{K_2})\ (\exists v \in V_{K_1}(w) \cap V_{K_2}(w)),\ \nexists u \in U : w \circ vu \in \overline{K_1} \cap \overline{K_2}. \tag{4.9}$$

Desenvolvendo-se (4.9) de forma semelhante a feita anteriormente, obtém-se:

$$(\exists w \in L((F_1 \wedge F_2)/H))\ (\exists v \in V_L(w))\ (\nexists u \in U_L(w, v))\ :\ w \circ vu \in L((F_1 \wedge F_2)/H).$$

Mas $w \circ vu \in L((F_1 \wedge F_2)/H)$ *implica* $vu \in (F_1 \wedge F_2)(w)$*, de onde se conclui que:*

$$(\exists w \in L((F_1 \wedge F_2)/H))\ (\exists v \in V_L(w))\ (\nexists u \in U_L(w, v))\ :\ vu \in (F_1 \wedge F_2)(w).$$

Portanto $(\exists w \in L((F_1 \wedge F_2)/H)) : (F_1 \wedge F_2)(w) \notin \Gamma(w)$*. Ou seja,* $F_1 \wedge F_2$ *não é consistente para* H*.*

$\diamond$

Proposição 4.4 *Sejam $K_1, K_2 \subseteq L$ duas linguagens não vazias, vu-controláveis e.r.a. L e interconsistentes, então $K_1 \cap K_2 \neq \emptyset$.*

Prova: *Supor que $K_1, K_2 \neq \emptyset$, mas que $K_1 \cap K_2 = \emptyset$. Na seqüência, mostra-se que uma das linguagens não é vu-controlável ou então que elas não são interconsistentes.*

a) Supor inicialmente que $K_1 = \{\epsilon\}$. Como $K_1 \cap K_2 = \emptyset$, então $\epsilon \notin K_2$. Mas se $K_2 \neq \emptyset$ e $K_2 \neq \epsilon$, então $V_{K_2}(\epsilon) \neq \emptyset$, sendo que $\epsilon \in \overline{K_2}$. Seja K_2 uma linguagem vu-controlável e.r.a. L, então para $\epsilon \in \overline{K_2}$ tem-se que $V_{K_2}(\epsilon) = V_L(\epsilon) \neq \emptyset$. Entretanto, para $K_1 = \epsilon$ tem-se que $V_{K_1}(\epsilon) = \emptyset \neq V_L(\epsilon)$, ou seja, K_1 não é vu-controlável e.r.a. L.

b) Supor agora que $K_1, K_2 \neq \{\epsilon\}$, sendo $K_1, K_2 \neq \emptyset$. Assim, para $\epsilon \in \overline{K_1} \cap \overline{K_2}$ tem-se que $V_{K_1}(\epsilon), V_{K_2}(\epsilon) \neq \emptyset$. Supondo que K_1 e K_2 são vu-controláveis e.r.a. L, tem-se que $(\forall w \in \overline{K_1}) V_{K_1}(w) = V_L(w)$ e $(\forall w \in \overline{K_2}) V_{K_2}(w) = V_L(w)$. Desta forma, para qualquer cadeia $w \in \overline{K_1} \cap \overline{K_2}$ tem-se que $V_{K_1}(w) = V_{K_2}(w) = V_L(w)$. Portanto, para $\epsilon \in \overline{K_1} \cap \overline{K_2}$ sabe-se que $V_L(\epsilon) = V_{K_1}(\epsilon) = V_{K_2}(\epsilon) \neq \emptyset$, de onde se conclui que $V_{K_1}(\epsilon) \cap V_{K_2}(\epsilon) \neq \emptyset$. Por fim, supondo que $K_1 \cap K_2 = \emptyset$, pode-se afirmar que $V_{K_1 \cap K_2}(\epsilon) = \emptyset$ e que portanto para $\epsilon \in \overline{K_1} \cap \overline{K_2}$ tem-se que $V_{K_1}(\epsilon) \cap V_{K_2}(\epsilon) \nsubseteq V_{K_1 \cap K_2}(\epsilon)$, ou seja, que K_1 e K_2 não são interconsistentes. ◇

Proposição 4.5 *Sejam $K_1, K_2 \subseteq (V^+ \times U)^*$. Se $Sup\mathbb{C}_{VU}(K_1)$ e $Sup\mathbb{C}_{VU}(K_2)$ são interconsistentes, então:*

$$Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2) = Sup\mathbb{C}_{VU}(K_1 \cap K_2).$$

Prova:

1. $Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2) \supseteq Sup\mathbb{C}_{VU}(K_1 \cap K_2)$.

   Para realizar esta parte da prova considera-se o fato de que $Sup\mathbb{C}_{VU}(K)$ é monotônico, ou seja, se $K' \subseteq K$ então $Sup\mathbb{C}_{VU}(K') \subseteq Sup\mathbb{C}_{VU}(K)$. Assim, como $K_1 \cap K_2 \subseteq K_1$ tem-se que $Sup\mathbb{C}_{VU}(K_1 \cap K_2) \subseteq Sup\mathbb{C}_{VU}(K_1)$ e, da mesma forma, como $K_1 \cap K_2 \subseteq K_2$ tem-se que $Sup\mathbb{C}_{VU}(K_1 \cap K_2) \subseteq Sup\mathbb{C}_{VU}(K_2)$. Desta forma, pode-se afirmar que $Sup\mathbb{C}_{VU}(K_1 \cap K_2) \subseteq Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2)$, independentemente de $Sup\mathbb{C}_{VU}(K_1)$ e $Sup\mathbb{C}_{VU}(K_2)$ serem interconsistentes.

2. $Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2) \subseteq Sup\mathbb{C}_{VU}(K_1 \cap K_2)$.

   Sabe-se que $Sup\mathbb{C}_{VU}(K_1) \subseteq K_1$ *e* $Sup\mathbb{C}_{VU}(K_2) \subseteq K_2$, *logo* $Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2) \subseteq K_1 \cap K_2$. *Agora, de acordo com a Proposição 4.2, se* $Sup\mathbb{C}_{VU}(K_1)$ *e* $Sup\mathbb{C}_{VU}(K_2)$ *são interconsistentes, então* $Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2)$ *é vu-controlável e.r.a.* L. *Portanto,* $Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2) \subseteq Sup\mathbb{C}_{VU}(K_1 \cap K_2)$. $\diamond$

Pode-se apresentar agora o teorema que estabelece o principal resultado desta seção.

Teorema 4.1 *Sejam uma planta* $H = (L, \Gamma)$ *e duas especificações,* $E_i \subseteq P_V(L)$, $(i = 1, 2)$. *Se* $Sup\mathbb{C}_{VU}[P_V^{-1}(E_1) \cap L]$ *e* $Sup\mathbb{C}_{VU}[P_V^{-1}(E_2) \cap L]$ *são interconsistentes e não vazias, então existem dois SMN* F_1 *e* F_2, *tais que:*

1. $P_V[L_m(F_i/H)] = Sup\mathbb{C}_V(E_i) \neq \emptyset$; *e*

2. $F_1 \wedge F_2$ *é um SMN e.r.a.* $E_1 \cap E_2$ *tal que* $P_V[L_m((F_1 \wedge F_2)/H)] = P_V[Sup\mathbb{C}_{VU}(P_V^{-1}(E_1) \cap L) \cap Sup\mathbb{C}_{VU}(P_V^{-1}(E_2) \cap L)] = Sup\mathbb{C}_V(E_1 \cap E_2) \neq \emptyset$.

Prova: *No intuito de simplificar a notação usada nesta prova, faz-se* $K_i = P_V^{-1}(E_i) \cap L$, *com* $i = 1, 2$.

1. *Supor que* $Sup\mathbb{C}_{VU}(K_i) \neq \emptyset$. *Então, de acordo com o Corolário 3.1, sabe-se que existe um SMN* F_i *para* H *tal que* $P_V[L_m(F_i/H)] = P_V[Sup\mathbb{C}_{VU}(K_i)] = Sup\mathbb{C}_V(E_i) \neq \emptyset$.

2. $P_V[L_m((F_1 \wedge F_2)/H)] = P_V[Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2)] = Sup\mathbb{C}_V(E_1 \cap E_2) \neq \emptyset$

   Supor que $Sup\mathbb{C}_{VU}(K_1)$ *e* $Sup\mathbb{C}_{VU}(K_2)$ *sejam interconsistentes. Sejam então* F_1, F_2 *SMN para* H *tais que* $L_m(F_i/H) = Sup\mathbb{C}_{VU}(K_i)$. *Pela Proposição 4.3 tem-se que* $L_m((F_1 \wedge F_2)/H) = L_m(F_1/H) \cap L_m(F_2/H) = Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2)$. *Logo* $P_V[L_m((F_1 \wedge F_2)/H)] = P_V[Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2)]$. *Agora, pelas proposições 4.4 e 4.5 sabe-se que para* $Sup\mathbb{C}_{VU}(K_1)$ *e* $Sup\mathbb{C}_{VU}(K_2)$ *não vazias e interconsistentes tem-se que* $Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2) = Sup\mathbb{C}_{VU}(K_1 \cap K_2) \neq \emptyset$. *Desta forma,* $P_V[L_m((F_1 \wedge F_2)/H)] = P_V[Sup\mathbb{C}_{VU}(K_1 \cap K_2)]$. *Seja então* $K = K_1 \cap K_2 = [P_V^{-1}(E_1) \cap L] \cap [P_V^{-1}(E_2) \cap L] = P_V^{-1}(E_1) \cap P_V^{-1}(E_2) \cap L$. *Como* $P_V^{-1}(E_1) \cap P_V^{-1}(E_2) = P_V^{-1}(E_1 \cap E_2)$ *(Lema 4.1), então* $K = K_1 \cap K_2 = P_V^{-1}(E_1 \cap E_2) \cap L$. *Seja ainda* $E = E_1 \cap E_2$, *então tem-se que* $K = P_V^{-1}(E) \cap L$. *Sabe-se que* $E_1, E_2 \subseteq P_V(L)$ *e então* $E = E_1 \cap E_2 \subseteq P_V(L)$.

Assim, de acordo com a Proposição 3.7, para $K = K_1 \cap K_2 = P_V^{-1}(E_1 \cap E_2) \cap L = P_V^{-1}(E) \cap L$ *tem-se que* $P_V[Sup\mathbb{C}_{VU}(K_1 \cap K_2)] = P_V[Sup\mathbb{C}_{VU}(K)] = Sup\mathbb{C}_V(E) = Sup\mathbb{C}_V(E_1 \cap E_2)$. *Além disso, ainda pela Proposição 3.7 sabe-se que* $Sup\mathbb{C}_{VU}(K) \neq \emptyset \Longrightarrow Sup\mathbb{C}_V(E) \neq \emptyset$ *e então como* $L_m((F_1 \wedge F_2)/H) = Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2) = Sup\mathbb{C}_{VU}(K_1 \cap K_2) \neq \emptyset$, *tem-se que* $P_V[L_m((F_1 \wedge F_2)/H)] = P_V[Sup\mathbb{C}_{VU}(K_1 \cap K_2)] = Sup\mathbb{C}_V(E_1 \cap E_2) \neq \emptyset$.

Resta mostrar ainda que $F_1 \wedge F_2$ *é um SMN e.r.a.* $E_1 \cap E_2$, *ou seja, que* $\overline{L_m((F_1 \wedge F_2)/H)} = L((F_1 \wedge F_2)/H)$, *onde* $L_m((F_1 \wedge F_2)/H) = L((F_1 \wedge F_2)/H) \cap P_V^{-1}(E_1 \cap E_2)$, *o que é feito a seguir. Sabe-se que* $\overline{L_m((F_1 \wedge F_2)/H)} = \overline{L_m(F_1/H) \cap L_m(F_2/H)} = \overline{Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2)}$. *Mas como* $Sup\mathbb{C}_{VU}(K_1)$ *e* $Sup\mathbb{C}_{VU}(K_2)$ *são interconsistentes, então (Proposição 4.1)* $\overline{Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2)} = \overline{Sup\mathbb{C}_{VU}(K_1)} \cap \overline{Sup\mathbb{C}_{VU}(K_2)}$. *Desta forma,* $\overline{L_m((F_1 \wedge F_2)/H)} = \overline{L_m(F_1/H)} \cap \overline{L_m(F_2/H)}$. *Agora, como os supervisores* F_i *são SMN e.r.a.* E_i, *então* $\overline{L_m(F_i/H)} = L(F_i/H)$, *onde* $L_m(F_i/H) = L(F_i/H) \cap P_V^{-1}(E_i)$. *Sendo assim,* $\overline{L_m((F_1 \wedge F_2)/H)} = L(F_1/H) \cap L(F_2/H) = L((F_1 \wedge F_2)/H)$. *Continuando, tem-se que* $L_m((F_1 \wedge F_2)/H) = L_m(F_1/H) \cap L_m(F_2/H) = [L(F_1/H) \cap P_V^{-1}(E_1)] \cap [L(F_2/H) \cap P_V^{-1}(E_2)] = [L(F_1/H) \cap L(F_2/H)] \cap [P_V^{-1}(E_1) \cap P_V^{-1}(E_2)] = L((F_1 \wedge F_2)/H) \cap [P_V^{-1}(E_1) \cap P_V^{-1}(E_2)]$. *Mas, pelo Lema 4.1 sabe-se que* $P_V^{-1}(E_1) \cap P_V^{-1}(E_2) = P_V^{-1}(E_1 \cap E_2)$, *de onde se obtém que* $L_m((F_1 \wedge F_2)/H) = L((F_1 \wedge F_2)/H) \cap P_V^{-1}(E_1 \cap E_2)$. *Sendo assim, pode-se afirmar que* $F_1 \wedge F_2$ *é um SMN e.r.a.* $E_1 \cap E_2$ *tal que* $P_V[L_m((F_1 \wedge F_2)/H)] = P_V[Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2)] = Sup\mathbb{C}_V(E_1 \cap E_2) \neq \emptyset$.

$\diamond$

Considere então que uma especificação $E \subseteq V^*$ para o comportamento da planta H em malha fechada possa ser subdividida em duas partes, ou seja, $E = E_1 \cap E_2$. Se $Sup\mathbb{C}_{VU}[P_V^{-1}(E_1) \cap L]$ e $Sup\mathbb{C}_{VU}[P_V^{-1}(E_2) \cap L]$ são interconsistentes, então existem supervisores F_i não bloqueantes para H que implementam as máximas linguagens v-controláveis contidas em E_i e cuja conjunção resulta numa ação de controle não bloqueante e ótima, no sentido de ser menos restritiva possível. Em particular, os supervisores construídos da forma mostrada no Teorema 4.1 possuem estas propriedades.

Proposição 4.6 *Sejam F_1 e F_2 supervisores para H e dados $\mathcal{F}_1$ e $\mathcal{F}_2$ supervisores para $\mathcal{H}$, tais que F_i e $\mathcal{F}_i$ são logicamente equivalentes, $i = 1, 2$. Então $F_1 \wedge F_2$ e $\mathcal{F}_1 \wedge \mathcal{F}_2$ são logicamente equivalentes, ou seja:*

$$L((F_1 \wedge F_2)/H) = \mathcal{L}((\mathcal{F}_1 \wedge \mathcal{F}_2)/\mathcal{H}) \ \ e \ \ L_m((F_1 \wedge F_2)/H) = \mathcal{L}_m((\mathcal{F}_1 \wedge \mathcal{F}_2)/\mathcal{H}).$$

Prova: *Sejam F_i e $\mathcal{F}_i$ supervisores logicamente equivalentes, onde $i = 1, 2$. Então $L(F_i/H) = \mathcal{L}(\mathcal{F}_i/\mathcal{H})$ e $L_m(F_i/H) = \mathcal{L}_m(\mathcal{F}_i/\mathcal{H})$. Assim $L((F_1 \wedge F_2)/H) = L(F_1/H) \cap L(F_2/H) = \mathcal{L}(\mathcal{F}_1/\mathcal{H}) \cap \mathcal{L}(\mathcal{F}_2/\mathcal{H}) = \mathcal{L}((\mathcal{F}_1 \wedge \mathcal{F}_2)/\mathcal{H})$. Da mesma forma, tem-se que $L_m((F_1 \wedge F_2)/H) = L_m(F_1/H) \cap L_m(F_2/H) = \mathcal{L}_m(\mathcal{F}_1/\mathcal{H}) \cap \mathcal{L}_m(\mathcal{F}_2/\mathcal{H}) = \mathcal{L}_m((\mathcal{F}_1 \wedge \mathcal{F}_2)/\mathcal{H})$.* ◇

Com base na Proposição 4.6 pode-se afirmar que $F_1 \wedge F_2$ é não bloqueante para H se e somente se $\mathcal{F}_1 \wedge \mathcal{F}_2$ é não bloqueante para $\mathcal{H}$.

A Proposição 4.6 indica que o problema de síntese de supervisores modulares para sistemas híbridos pode ser solucionado por intermédio da resolução de um problema equivalente no domínio de SEDs.

Corolário 4.1 *Seja H o modelo de sistemas a eventos discretos para a planta híbrida $\mathcal{H}$ e sejam os supervisores F_i obtidos de acordo com o Teorema 4.1, com $i = 1, 2$. Se as linguagens implementadas por estes supervisores são interconsistentes, então os supervisores condição/evento $\mathcal{F}_i$ equivalentes aos supervisores F_i levam a uma solução ótima para o problema de síntese modular de supervisores para sistemas híbridos.*

Prova: *Segue diretamente do Teorema 4.1 e da Proposição 4.6.* ◇

A resolução de um problema de síntese de supervisores modulares para sistemas híbridos, considerando-se conhecido o modelo lógico para este sistema, é obtida pela utilização do seguinte procedimento.

1. Dadas duas especificações $E_1, E_2 \subseteq V^*$, obter as especificações equivalentes $K_i \subseteq L \subseteq (V^+ \times U)^*$ tais que $P_V(K_i) = E_i$, onde $i = 1, 2$;
2. Verificar se cada linguagem K_i é *vu*-controlável em relação a L. Se for, pular para o passo 3. Caso contrário, deve-se obter a máxima linguagem *vu*-controlável contida em K_i (apenas para K_i não controlável), denotada por $Sup\mathbb{C}_{VU}(K_i)$;

3. Testar se $Sup\mathbb{C}_{VU}(K_1)$ e $Sup\mathbb{C}_{VU}(K_2)$ são interconsistentes. Se forem, então os supervisores F_i são implementados por intermédio de autômatos tais que $L_m(F_i/H) = Sup\mathbb{C}_{VU}(K_i)$. Assim, de acordo com o Teorema 4.1, a ação conjunta dos supervisores modulares será não bloqueante e minimamente restritiva.

   Pode-se determinar ainda a máxima linguagem v-controlável contida em $E_1 \cap E_2$, fazendo $Sup\mathbb{C}_V(E_1 \cap E_2) = P_V[Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2)]$.

Conforme explicado no Capítulo 3, o projeto individual dos supervisores modulares pode ser feito com auxílio do GRAIL. Além disso, o teste da interconsistência também pode ser feito com auxílio do GRAIL. Esta última função foi desenvolvida no contexto do trabalho (Rodrigues, 2004), onde se pode encontrar o algoritmo usado para a verificação da interconsistência.

A seguir, apresenta-se um exemplo simples a fim de ilustrar a metodologia de resolução de problemas de controle supervisório modular de sistemas híbridos. Este mesmo exemplo foi utilizado no Capítulo 3 (Exemplo 3.7) para ilustrar a resolução através da abordagem monolítica e é retomado aqui de forma a possibilitar uma comparação dos resultados obtidos. Considera-se conhecido o modelo que representa o comportamento lógico para o sistema híbrido a ser controlado.

Exemplo 4.3 *Seja um sistema híbrido $\mathcal{H}_1$ cujo comportamento lógico $L_1 = \mathcal{L}(\mathcal{H}_1)$ é reconhecido pelo autômato mostrado na Figura 4.6.*

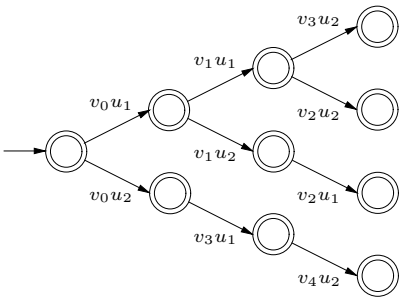
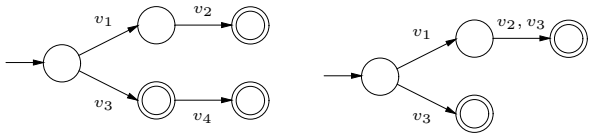


Figura 4.6: Planta Livre.

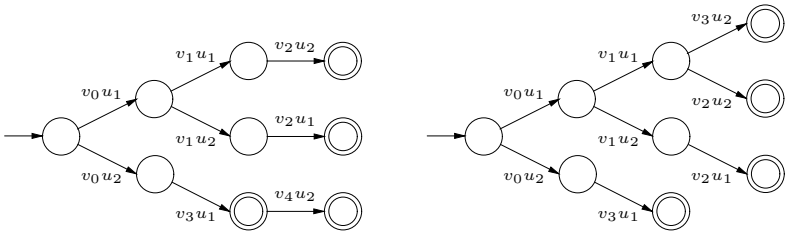
Sejam ainda duas especificações $E_1, E_2 \subseteq P_V(L_1)$ ilustradas pelos autômatos da Figura 4.7.



(a) $E_1 \subseteq P_V(L_1)$ (b) $E_2 \subseteq P_V(L_1)$

Figura 4.7: Especificações $E_1, E_2 \subseteq P_V(L_1)$.

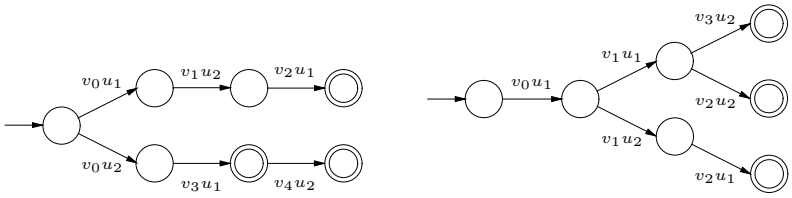
O primeiro passo para a síntese de supervisores modulares consiste em obter especificações equivalentes que estejam contidas na linguagem da planta. Estas especificações são obtidas fazendo-se $K_i = P_V^{-1}(E_i) \cap L_1$, sendo $i = 1, 2$. Autômatos que reconhecem tais linguagens são mostrados na Figura 4.8.



(a) $K_1 = P_V^{-1}(E_1) \cap L_1$. (b) $K_2 = P_V^{-1}(E_2) \cap L_1$.

Figura 4.8: Especificações $K_1, K_2 \subseteq (V^+ \times U)^*$.

Verifica-se, entretanto, que K_1 e K_2 não são vu-controláveis e.r.a. L_1. Obtêm-se então as máximas linguagens vu-controláveis contidas nestas especificações, ou seja, obtêm-se $Sup\mathbb{C}_{VU}(K_1)$ e $Sup\mathbb{C}_{VU}(K_2)$. Na figura 4.9 apresentam-se autômatos que reconhecem estas linguagens.



(a) $Sup\mathbb{C}_{VU}(K_1)$. (b) $Sup\mathbb{C}_{VU}(K_2)$.

Figura 4.9: Máximas Linguagens vu-controláveis.

Neste momento deve-se testar se $Sup\mathbb{C}_{VU}(K_1)$ e $Sup\mathbb{C}_{VU}(K_2)$ são interconsistentes. Neste caso tal condição é verificada e então os supervisores podem ser implementados de forma modular. Vale lembrar que os supervisores F_1 e F_2 podem ser implementados pelos autômatos mostrados na Figuras 4.9(a) e 4.9(b), respectivamente.

Pode-se determinar agora a máxima linguagem vu-controlável resultante da ação conjunta dos supervisores modulares. Esta linguagem é obtida fazendo-se $L_m((F_1 \wedge F_2)/H) = Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2)$ *e um autômato que reconhece a mesma é mostrado na Figura 4.10.*

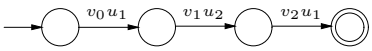


Figura 4.10: Linguagem vu-controlável obtida com $F_1 \wedge F_2$.

Por fim, a linguagem v-controlável resultante da ação conjunta dos dois supervisores é obtida fazendo-se $P_V[L_m((F_1 \wedge F_2)/H)] = P_V[Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2)]$. *Um autômato que reconhece esta linguagem é mostrado na Figura 4.11.*

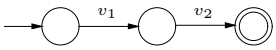


Figura 4.11: Linguagem v-controlável obtida com $F_1 \wedge F_2$.

Comparando este resultado com o obtido na abordagem monolítica, verifica-se que $P_V[L_m((F_1 \wedge F_2)/H)] = P_V[Sup\mathbb{C}_{VU}(K_1) \cap Sup\mathbb{C}_{VU}(K_2)] = Sup\mathbb{C}_V(E_1 \cap E_2) = v_1 \circ v_2$, *isto é, a solução obtida na abordagem modular é igual àquela obtida na abordagem monolítica (vide Exemplo 3.7).* ■

4.4 Exemplo

Nesta seção utiliza-se um exemplo de forma a ilustrar a abordagem proposta para a síntese de supervisores modulares para sistemas híbridos. O sistema em questão consiste de 3 trens que trafegam sobre trilhos cíclicos com trechos compartilhados, conforme mostrado na Figura 4.12. Este exemplo foi inicialmente apresentado em (Leal e Cury, 2004a) e depois foi levemente modificado em (Leal e Cury, 2004b), chegando ao problema apresentado aqui.

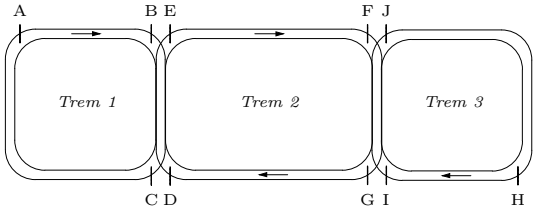


Figura 4.12: Três trens sobre trilhos cíclicos.

Neste sistema, existem sensores sobre os trilhos que registram a passagem dos trens pelas posições A, B e C (trem 1); D, E, F e G (trem 2); e H, I e J (trem 3). O trem 2 pode trafegar em dois modos de velocidade: *rápido* e *devagar*, e os trens 1 e 3 podem trafegar somente no modo *rápido*. Entretanto, a velocidade do trem 2 pode ser alterada somente nos instantes em que o mesmo passa pelos pontos D, E, F e G. Uma característica importante deste sistema é que o trem 2 pode viajar no modo *devagar* somente nos trechos $\overline{EF}$ e $\overline{GD}$, ou seja, o modo devagar pode ser comandado somente quando o trem 2 cruza pelos pontos E e G. Esta limitação implica restrição nas possibilidades de chaveamento. Além disso, ao atingir o ponto E, o trem 2 pode parar, mas uma vez parado ele não pode mais voltar a trafegar. Esta possibilidade de parada do trem 2 é considerada com o intuito de utilizar marcação de linguagem, conforme mostrado ao longo da resolução do exemplo. O objetivo consiste em evitar colisões entre os trens e também garantir que o trem 2 não pare.

O comprimento de cada trilho e a localização dos sensores é como mostrado na Figura 4.13. Os pontos A, D e H foram escolhidos como referências para a identificação do posicionamento dos trens sobre os respectivos trilhos. No início, o trem 1 está 68 km depois de A (trecho $\overline{CA}$), o trem 2 está 13 km após D (trecho $\overline{DE}$), e o trem 3 está 11 km depois de H (trecho $\overline{HI}$). O valor para a velocidade no modo *devagar* (trem 2) é $\dot{x}_2 = 30\,\text{km/h}$ e para o modo *rápido* é $\dot{x}_i = 60\,\text{km/h}$, $i = 1, 2, 3$.

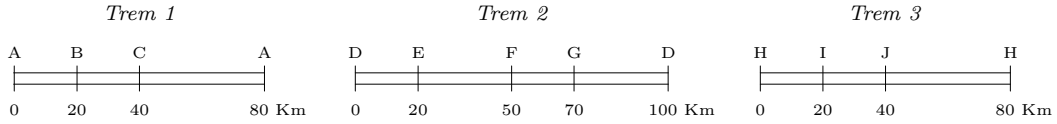


Figura 4.13: Trilhos cíclicos.

No intuito de não introduzir saltos na trajetória contínua de estados, modela-se a variável posição de forma que a mesma assuma valores apenas no intervalo $[0, max_i/2]$ (ver Figura 4.14), onde max_i é o comprimento do trilho em que o trem i trafega.

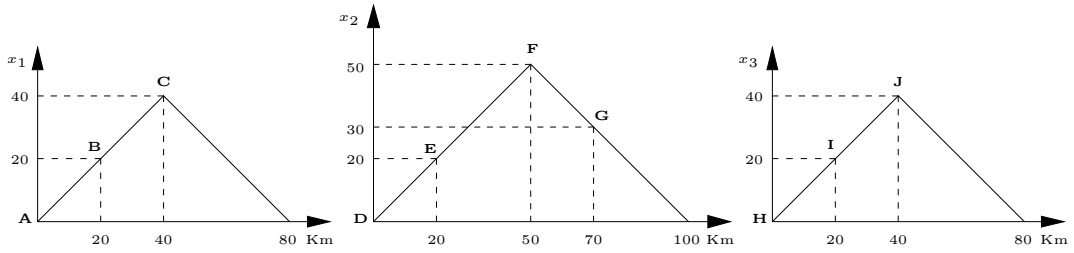
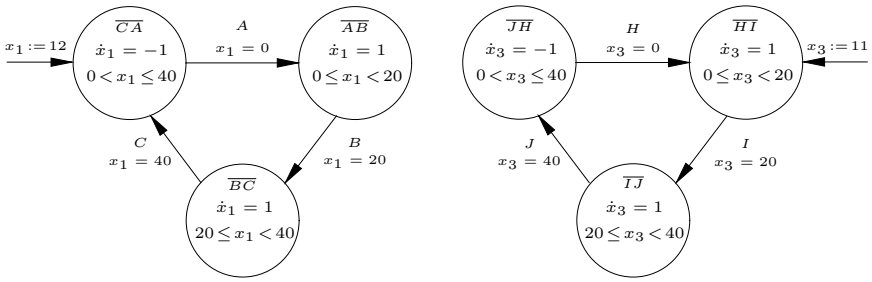
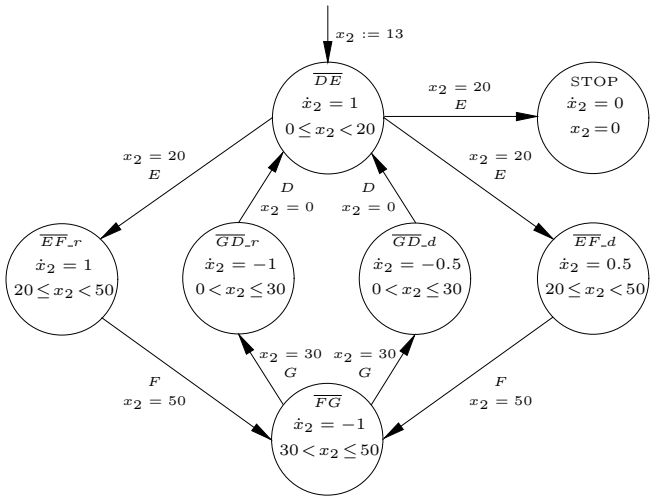


Figura 4.14: Posição x_i como uma função da localização do trem i no respectivo trilho.

A Figura 4.15 apresenta autômatos híbridos (Alur et al., 1993) que modelam o comportamento de cada trem que compõe o sistema. Observe que o modelo para o trem 2 é não determinístico: no estado $\overline{DE}$, a ocorrência do evento E ($x_2 = 20$) pode levar o trem 2 a parar (estado STOP), a adotar o modo rápido (estado $\overline{EF}\_r$) ou a adotar o modo devagar (estado $\overline{EF}\_d$) de velocidade. Deve-se ressaltar que as dinâmicas contínuas apresentadas nos autômatos híbridos da Figura 4.15 são dadas em km/min e não em km/h. Assim, $\dot{x}_i = 1$ significa que o trem i trafega a uma velocidade de 1 km/min, o que equivale a uma velocidade de 60 km/h.



(a) Trem 1 (b) Trem 3



(c) Trem 2

Figura 4.15: Autômatos híbridos para o exemplo dos 3 trens.

Os seguintes passos são executados para a resolução do problema:

Passo 1: Construir o modelo para o sistema híbrido.

Para a construção do modelo utiliza-se o CheckMate[1] (Chutinan, 1999). O espaço de estados contínuo é dado por $x = [x_1 \quad x_2 \quad x_3]^{\mathrm{T}}$, onde x_i é a posição do trem i ($i = 1, 2, 3$), medida a partir da origem. O cruzamento de um trem por um sensor de posição é interpretado como a ocorrência de um evento de limiar. Com o intuito de simplificar o modelo, considera-se que não ocorrem cruzamentos simultâneos de trens por sensores de posição e então tem-se um total de 10 eventos de limiar diferentes (a ocorrência simultânea de eventos pode ser interpretada como uma seqüência de eventos). Além disso, utilizam-se símbolos para representar os eventos de limiar. O símbolo A, por exemplo, representa o evento $v(t) = [1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]^{\mathrm{T}}$, o qual ocorre naqueles instantes em que o trem 1 cruza pela posição A. Assim, $V = \{A, B, C, D, E, F, G, H, I, J\}$.

Passo 2: Obter um autômato C/E que represente o comportamento lógico aproximado do sistema híbrido.

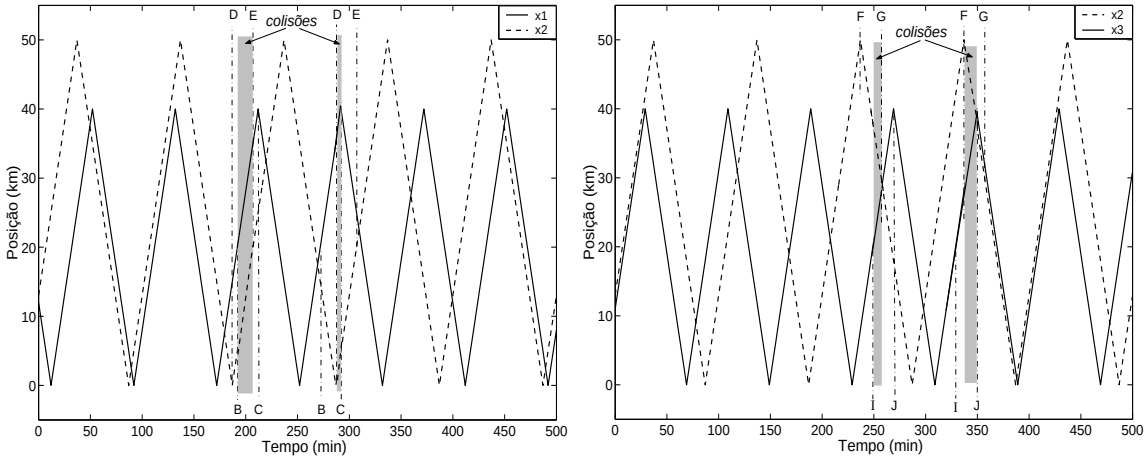
O cálculo de uma aproximação de estados finitos para a planta híbrida é feito através do CheckMate. O autômato obtido é então convertido em um autômato C/E dado em um formato compatível com uma ferramenta computacional chamada Grail (Raymond e Wood, 1995). Esta conversão é realizada por uma função desenvolvida para o Matlab (*cm2grail.m*) com este propósito. Para este exemplo, obteve-se uma máquina de estados finitos com 238 estados e 264 transições, a qual (neste caso) modela o comportamento lógico exato do sistema em malha aberta (sem supervisão). Por uma questão de espaço e legibilidade, não se mostra aqui este modelo.

Passo 3: Sintetizar os supervisores modulares.

Através da simulação do comportamento do sistema em malha aberta, na qual o modo *rápido* de velocidade é sempre escolhido, verifica-se a existência de colisões entre os trens nos dois trechos compartilhados. Na Figura 4.16, mostrada abaixo, as regiões sombreadas indicam a ocorrência de colisões: observe que o trem 2 cruza pela posição D e que antes dele cruzar pela posição E (sair do trecho $\overline{DE}$) o trem 1 cruza pela posição B (entra no trecho $\overline{BC}$),

[1] Desenvolvida no ECE (Electrical and Computer Engineering Department) da Universidade de Carnegie Mellon (CMU), o CheckMate é uma ferramenta baseada no Matlab que serve para a simulação e verificação de sistemas híbridos.

ou seja, neste intervalo ambos os trens trafegam no trecho compartilhado, o que leva a uma colisão entre os mesmos.



(a) Trens 1 e 2 (b) Trens 2 e 3

Figura 4.16: Comportamento do sistema em malha aberta.

No intuito de evitar colisões entre os trens, propõe-se a utilização de dois supervisores modulares, um para evitar colisões entre os trens 1 e 2 (especificação E_1) e outro para evitar colisões entre os trens 2 e 3 (especificação E_2). A especificação referente a não parada do trem 2 é obtida marcando-se apenas o estado inicial de E_1, o que pode ser interpretado como uma especificação para garantir que, após o trem 2 sensibilizar o sensor E, uma nova ocorrência do evento D sempre poderá acontecer. As especificações $E_1, E_2 \subseteq V^*$ são mostradas na Figura 4.17.

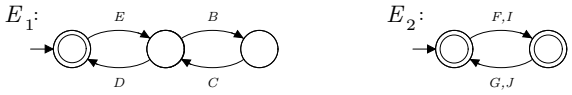


Figura 4.17: Especificações $E_1, E_2 \subseteq V^*$.

Seguindo o procedimento de síntese de supervisores descrito no Capítulo 3, e usando uma ferramenta de controle supervisório desenvolvida em (González, 2000), foram obtidos um supervisor F_1 com 92 estados e 97 transições e um supervisor F_2 com 97 estados e 106 transições. Usando uma função desenvolvida no contexto deste trabalho, pode-se verificar que as linguagens implementadas por estes supervisores são *interconsistentes*. Sendo assim, de acordo com o Teorema 4.1, a ação conjunta destes supervisores é ótima (não bloqueante e minimamente restritiva). A Figura 4.18 mostra o autômato que representa $F_1 \wedge F_2$, na qual $u_1 = [r_1 \quad r_2 \quad r_3]^{\mathrm{T}}$

e $u_2 = [r_1 \quad d_2 \quad r_3]^{\mathrm{T}}$, onde r_i refere-se ao modo *rápido* e d_i ao modo *devagar* de velocidade para o trem i. Note que foi realizada uma agregação de estados de forma a mostrar apenas os eventos associados ao trem 2, uma vez que este é o único que pode ter sua velocidade modificada.

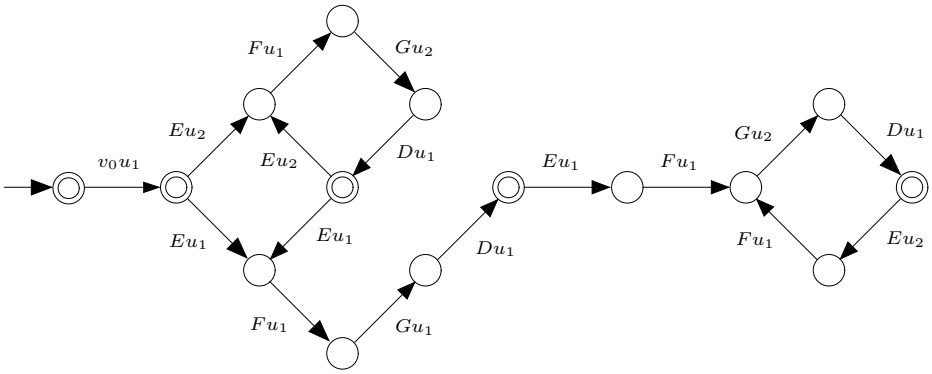
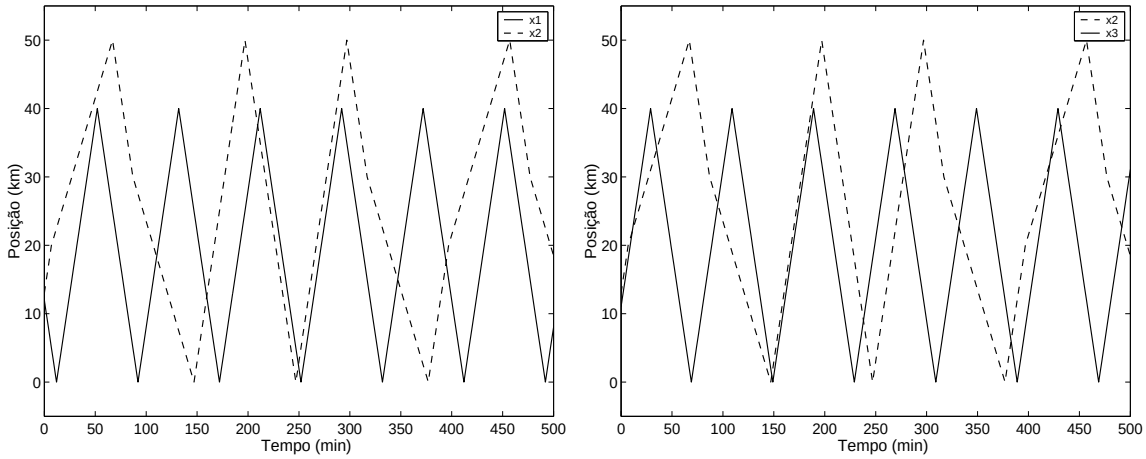


Figura 4.18: Supervisor agregado $F_1 \wedge F_2$.

A Figura 4.19 apresentada a seguir, mostra o resultado de uma simulação para o comportamento da planta sob a ação conjunta dos supervisores modulares. Pode-se verificar que agora não ocorrem mais colisões entre os trens, conforme era de se esperar. Nesta simulação, a seqüência de pares (v, u) adotada pelo sistema sob supervisão é dada por $v_0u_1 \circ Eu_2 \circ Fu_1 \circ Gu_2 \circ Du_1 \circ Eu_1 \circ Fu_1 \circ Gu_1 \circ Du_1 \circ Eu_1 \circ Fu_1 \circ (Gu_2 \circ Du_1 \circ Eu_2 \circ Fu_1)^*$.



(a) Trens 1 e 2 (b) Trens 2 e 3

Figura 4.19: Comportamento do sistema sob supervisão modular.

Por fim, de forma a solucionar o problema original, os supervisores SED encontrados no passo 3 podem ser usados como implementações discretas dos supervisores C/E.

4.5 Conclusões

Neste capítulo, estende-se a abordagem de controle supervisório para sistemas híbridos, apresentada no capítulo anterior, pela introdução de uma abordagem modular para a síntese dos supervisores. A classe de sistemas híbridos considerada é a mesma do Capítulo 3, ou seja, uma classe de sistemas que combinam dinâmicas contínuas e dinâmicas discretas, na qual eventos de limiar (gerados quando variáveis de estado alcançam certos patamares) forçam transição de estado discreto e cujas dinâmicas contínuas são determinadas por uma condição discreta que depende do estado discreto atual do sistema.

Na síntese de supervisores modulares, ao invés de projetar um único supervisor monolítico que satisfaça todas as especificações, procura-se construir um supervisor para cada especificação (ou para um pequeno conjunto de especificações), de forma que, atuando em conjunto, os supervisores satisfaçam a especificação global. Sendo assim, a síntese modular permite que problemas complexos possam ser decompostos em problemas mais simples.

Verificou-se neste trabalho que, diferentemente da teoria clássica de controle modular de SEDs, para os sistemas híbridos a modularidade entre as especificações não é suficiente para garantir que a ação conjunta dos supervisores modulares leve o sistema sob supervisão a ter um comportamento não bloqueante e minimamente restritivo, conforme desejado. Introduziu-se então o conceito de *interconsistência* entre linguagens e estabeleceram-se condições sob as quais pode-se utilizar a abordagem modular para o controle supervisório de sistemas híbridos.

Propôs-se, assim, um procedimento para realização de síntese de supervisores para sistemas híbridos utilizando uma abordagem modular e um exemplo foi apresentado de forma a ilustrar esta metodologia. Neste exemplo mostrou-se o procedimento desde a modelagem feita no ambiente Simulink/Matlab com emprego da ferramenta CheckMate, passando pela obtenção de um autômato aproximação para o comportamento lógico do sistema híbrido, obtido por intermédio do CheckMate, resolvendo o problema no domínio de SEDs com auxílio do Grail e, por fim, por intermédio do Simulink, fazendo uma simulação do comportamento do sistema em malha fechada. Todo processo de síntese dos supervisores modulares foi feito com auxílio da ferramenta Grail, utilizando-se funções desenvolvidas especificamente para este propósito.

A síntese modular de supervisores para sistemas híbridos foi tratada anteriormente apenas em (Moor, Davoren e Raisch, 2001). Naquele trabalho o problema é formulado usando a

teoria comportamental de sistemas, introduzida por Willems (1991). Já no presente trabalho o problema é apresentado através do formalismo de sistemas condição/evento (Sreenivas e Krogh, 1991), o qual leva a determinados benefícios. Uma diferença importante entre estes trabalhos consiste no conceito de bloqueio usado em cada um deles. Em (Moor, Davoren e Raisch, 2001) o bloqueio corresponde a "deadlock", enquanto que aqui o bloqueio está relacionado à impossibilidade de completar uma tarefa, sendo este último um conceito mais geral que o anterior. Ainda, um conceito importante introduzido neste trabalho, e que não possui um equivalente naquele outro, é o conceito de *supervisor marcador*. Este conceito permite que a determinação do que deve ser uma tarefa fique a cargo do supervisor – e portanto a cargo da especificação feita pelo projetista. Além disso, em (Moor, Davoren e Raisch, 2001), bem como nos demais trabalhos daqueles autores, não é explicitado o procedimento de síntese de supervisores, o que torna a utilização da metodologia uma tarefa extremamente difícil. Ainda, enquanto neste trabalho todo o procedimento de síntese de supervisores (passando pela obtenção do modelo da planta) é feito com auxílio de um programa mundialmente comercializado (MATLAB) e de uma ferramenta gratuita disponível na Internet (GRAIL), no trabalho de Moor, Davoren e Raisch (2001) nada é dito sobre alguma ferramenta para obtenção do modelo lógico para a planta híbrida e a ferramenta citada para síntese do supervisor não está disponível na Internet. Entretanto, a maior contribuição deste trabalho é a garantia de sempre obter supervisores modulares que levam a uma solução ótima, contribuição esta que será apresentada no capítulo subseqüente.

Capítulo 5

Existência de Soluções Ótimas para a Abordagem Modular

5.1 Introdução

A partir dos resultados estabelecidos no capítulo anterior, conclui-se que, se na síntese modular de supervisores para sistemas híbridos as linguagens $Sup\mathbb{C}_{VU}[P_V^{-1}(E_1) \cap L]$ e $Sup\mathbb{C}_{VU}[P_V^{-1}(E_2) \cap L]$ são interconsistentes, então os supervisores que implementam individualmente estas linguagens em malha fechada são tais que sua ação conjunta é não bloqueante e minimamente restritiva. A pergunta que surge então é: "– Mas e se estas linguagens não são interconsistentes?" Este problema é tratado ao longo deste capítulo, tendo sido abordado previamente em (Leal e Cury, 2004c).

5.2 Resolução de Conflitos

Da mesma forma que na teoria clássica de controle modular de SEDs, quando a síntese de supervisores modulares para sistemas híbridos leva a um conflito (linguagens não interconsistentes), deve-se buscar a solução deste conflito para que se possam utilizar os supervisores de forma modular. Entretanto, conforme mostrado na seqüência, a grande vantagem da abordagem proposta neste trabalho é que a mesma sempre levará a uma solução ótima, ou seja, se existe uma solução para um problema de síntese monolítica de supervisores para sistemas híbridos, então o mesmo resultado certamente será obtido através da abordagem modular.

Seja $\mathcal{I}(K_1, K_2)$ o conjunto de todas as sublinguagens de K_1 que são interconsistentes em relação a K_2. Inicialmente mostra-se que um elemento supremo sempre existe neste conjunto.

Proposição 5.1 *$\mathcal{I}(K_1, K_2)$ é não vazio e fechado sob uniões arbitrárias. Em particular, $\mathcal{I}(K_1, K_2)$ contém um (único) elemento supremo, o qual denota-se por $Sup\mathcal{I}(K_1, K_2)$.*

Prova: *Uma vez que $\emptyset \in \mathcal{I}(K_1, K_2)$ por definição, então esta classe é não vazia.*

Sejam K_a e K_b duas linguagens tais que $K_a, K_b \in \mathcal{I}(K_1, K_2)$. Inicialmente mostra-se que $K_a \cup K_b \in \mathcal{I}(K_1, K_2)$, ou seja, mostra-se que $K_a \cup K_b \subseteq K_1$ e que $(\forall w \in \overline{K_a \cup K_b} \cap \overline{K_2}) V_{K_a \cup K_b}(w) \cap V_{K_2}(w) = V_{(K_a \cup K_b) \cap K_2}(w)$. Antes, porém, mostra-se que $(\forall w \in \overline{K_a \cup K_b}) V_{K_a}(w) \cup V_{K_b}(w) = V_{K_a \cup K_b}(w)$. Seja $w \in \overline{K_a} \cup \overline{K_b}$, então $V_{K_a}(w) \cup V_{K_b}(w) = \{v \in V^+ : (\exists u \in U) w \circ vu \in \overline{K_a}\} \cup \{v \in V^+ : (\exists u \in U) w \circ vu \in \overline{K_b}\} = \{v \in V^+ : (\exists u \in U) w \circ vu \in \overline{K_a} \cup \overline{K_b}\}$. Mas $\overline{K_a \cup K_b} = \overline{K_a} \cup \overline{K_b}$, e então pode-se afirmar que $(\forall w \in \overline{K_a \cup K_b})\ V_{K_a}(w) \cup V_{K_b}(w) = \{v \in V^+ : (\exists u \in U) w \circ vu \in \overline{K_a \cup K_b}\} = V_{K_a \cup K_b}(w)$.

Continuando, de acordo com o mostrado acima, para $w \in \overline{K_a \cup K_b} \cap \overline{K_2}$ tem-se que $V_{K_a \cup K_b}(w) \cap V_{K_2}(w) = (V_{K_a}(w) \cup V_{K_b}(w)) \cap V_{K_2}(w) = (V_{K_a}(w) \cap V_{K_2}(w)) \cup (V_{K_b}(w) \cap V_{K_2}(w))$. Agora, como K_a e K_2 são interconsistentes, então $(\forall w \in \overline{K_a} \cap \overline{K_2}) V_{K_a}(w) \cap V_{K_2}(w) = V_{K_a \cap K_2}(w)$. Da mesma forma, como K_b e K_2 são interconsistentes, então $(\forall w \in \overline{K_b} \cap \overline{K_2}) V_{K_b}(w) \cap V_{K_2}(w) = V_{K_b \cap K_2}(w)$. Sendo assim, para $w \in \overline{K_a \cup K_b} \cap \overline{K_2}$ tem-se que $V_{K_a \cup K_b}(w) \cap V_{K_2}(w) = V_{K_a \cap K_2}(w) \cup V_{K_b \cap K_2}(w) = V_{(K_a \cap K_2) \cup (K_b \cap K_2)}(w) = V_{(K_a \cup K_b) \cap K_2}(w)$. Logo, $(K_a \cup K_b)$ e K_2 são interconsistentes.

Por fim, sabe-se que $K_a, K_b \in \mathcal{I}(K_1, K_2) \Rightarrow K_a, K_b \subseteq K_1$ e que portanto $K_a \cup K_b \subseteq K_1$. Desta forma, pode-se afirmar que $K_a \cup K_b \in \mathcal{I}(K_1, K_2)$.

Seja agora $K_\alpha \in \mathcal{I}(K_1, K_2)$ para todo α pertencente a um conjunto de índices A, e seja $K = \cup\{K_\alpha | \alpha \in A\}$. Estendendo-se a prova feita antes para um número arbitrário de linguagens, tem-se que $(\forall w \in \overline{\cup_{\alpha \in A} K_\alpha} \cap \overline{K_2})\ V_{\cup_{\alpha \in A} K_\alpha}(w) \cap V_{K_2}(w) = (\cup_{\alpha \in A} V_{K_\alpha}(w)) \cap V_{K_2}(w) = \cup_{\alpha \in A}(V_{K_\alpha}(w) \cap V_{K_2}(w))$. Mas se K_α e K_2 são interconsistentes para todo $\alpha \in A$, então $\cup_{\alpha \in A}(V_{K_\alpha}(w) \cap V_{K_2}(w)) = \cup_{\alpha \in A} V_{K_\alpha \cap K_2}(w)$. Assim, $(\forall w \in \overline{\cup_{\alpha \in A} K_\alpha} \cap \overline{K_2})\ V_{\cup_{\alpha \in A} K_\alpha}(w) \cap V_{K_2}(w) = \cup_{\alpha \in A} V_{K_\alpha \cap K_2}(w) = V_{\cup_{\alpha \in A}(K_\alpha \cap K_2)}(w) = V_{(\cup_{\alpha \in A} K_\alpha) \cap K_2}(w)$, ou ainda, $(\forall w \in \overline{K} \cap \overline{K_2})\ V_K(w) \cap V_{K_2}(w) = V_{K \cap K_2}(w)$. Além disso, sabe-se que $\cup_{\alpha \in A} K_\alpha \subseteq K_1$ e que portanto $K \subseteq K_1$. Logo, $\cup_{\alpha \in A} K_\alpha \in \mathcal{I}(K_1, K_2)$, isto é, $K \in \mathcal{I}(K_1, K_2)$. Finalmente, $Sup\mathcal{I}(K_1, K_2) = \cup\{K | K \in \mathcal{I}(K_1, K_2)\}$.

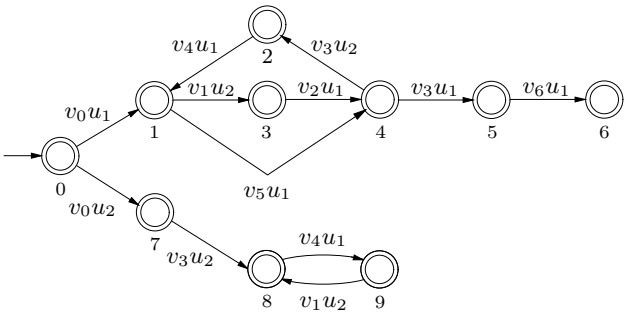
$\diamond$

O algoritmo para determinação de $Sup\mathcal{I}$ (Rodrigues, 2004) pode ser resumido nos seguintes passos:

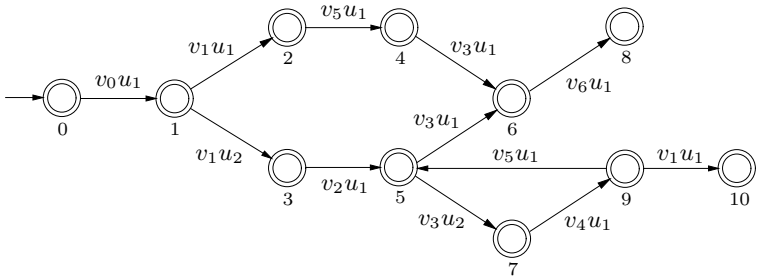
1. Obter um autômato para $K = K_1||K_2$;
2. Criar um autômato K_1^e como uma cópia do autômato para K e acrescentar estados e transições à K_1^e de forma que ele reconheça a linguagem K_1;
3. Determinar e apagar (de forma iterativa) os estados de K para os quais o teste da interconsistência falha, relacionando-os numa lista de "maus estados";
4. Apagar de K_1^e os maus estados listados no passo anterior;
5. Obter a componente acessível e determinizar o autômato resultante.

No exemplo apresentado a seguir ilustra-se o procedimento a ser adotado para o cálculo de $Sup\mathcal{I}$.

Exemplo 5.1 *Sejam K_1 e K_2 as linguagens reconhecidas pelos autômatos mostrados na Figura 5.1.*



(a) Autômato para K_1



(b) Autômato para K_2

Figura 5.1: Linguagens K_1 e K_2 para ilustrar o cálculo de $Sup\mathcal{I}(K_1, K_2)$.

O primeiro passo no cálculo de $Sup\mathcal{I}(K_1, K_2)$ consiste em obter um autômato para $K = K_1||K_2$. Este autômato é mostrado na Figura 5.2.

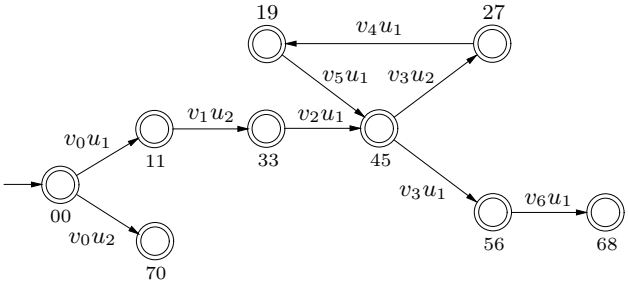


Figura 5.2: Autômato para $K = K_1||K_2$.

No segundo passo obtém-se um autômato K_1^e como uma cópia do autômato para K e acrescenta-se estados e transições de forma que K_1^e reconheça a linguagem K_1. Assim, se no estado q do autômato de K_1 existe uma transição vu que não está num estado (q, y) do autômato para K_1^e, então deve-se acrescentar esta transição no autômato para K_1^e. Por outro lado, se não existe um estado $(q, \cdot)$ em K_1^e, deve-se criá-lo, acrescentando também a referida transição. Repete-se o procedimento para todos os estados de K_1^e e para os novos estados criados. A diferença entre os autômatos para K_1 e para K_1^e é que o autômato para K_1^e não possui ambigüidade na determinação de maus estados. A título de exemplo, observe que no estado 1 do autômato para K_1 existe uma transição v_1u_2 que leva ao estado 3 e que no estado $(1, 9)$ do autômato para K esta transição não aparece. Assim, acrescenta-se a transição v_1u_2 do estado $(1, 9)$ ao estado $(3, 3)$ do autômato para K_1^e. O autômato resultante para K_1^e é mostrado na Figura 5.3.

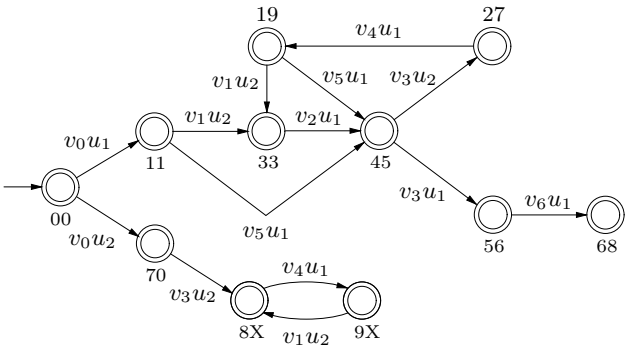
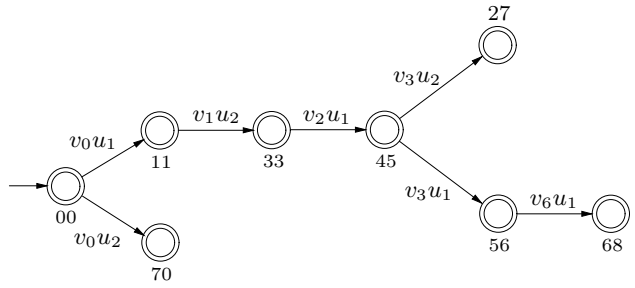


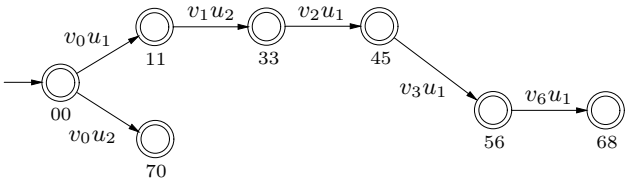
Figura 5.3: Autômato para K_1^e.

No terceiro passo, determina-se e apaga-se (de forma iterativa) os maus estados de K e relaciona-se estes maus estados numa lista. A determinação dos maus estados é feita da seguinte forma. Para cada estado xy de K verifica-se se o conjunto de eventos ativos no estado x de K_1 que está ativo no estado y de K_2 também está ativo no estado xy de K. Caso isto não seja verificado, o estado xy é considerado um mau estado. Este teste corresponde a verificar

se $(\forall w \in \overline{K_1} \cap \overline{K_2})\ V_{K_1}(w) \cap V_{K_2}(w) = V_K(w)$. *Na primeira iteração o estado* 19 *de* K *é o único mau estado, sendo retirado de* K *juntamente com as transições* v_4u_1 *e* v_5u_1. *A Figura 5.4(a) mostra o resultado obtido na primeira iteração. Este procedimento é iterativo, uma vez que um estado pode se tornar mau estado após a eliminação de outro numa iteração prévia. Observe que após a eliminação do estado* 19 *na primeira iteração, o estado* 27 *tornou-se um mau estado, sendo retirado então na segunda iteração juntamente com a transição* v_3u_2 *(ver Figura 5.4(b)). Assim,* 19 *e* 27 *são os maus estados de* K.



(a) Autômato para K após primeira iteração



(b) Autômato para K após segunda iteração

Figura 5.4: Resultados obtidos após eliminação de maus estados de $K = K_1||K_2$.

O quarto passo no cálculo de $Sup\mathcal{I}(K_1, K_2)$ *consiste em apagar de* K_1^e *os maus estados listados no passo anterior. São apagados então os estados* 19 *e* 27 *e as transições que saem e chegam nestes estados, obtendo-se assim o autômato mostrado na Figura 5.5.*

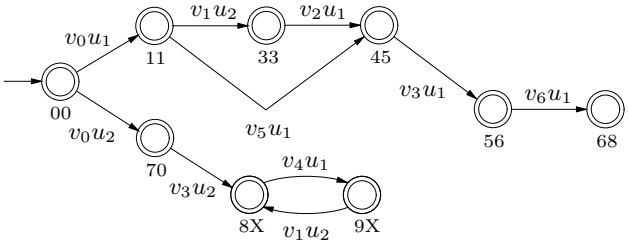


Figura 5.5: Autômato para $Sup\mathcal{I}(K_1, K_2)$.

Por fim, deve-se extrair a componente acessível e depois fazer a determinização do autômato obtido no passo anterior, uma vez que o mesmo pode ser não acessível e não determinístico. Neste exemplo, o autômato mostrado na Figura 5.5 representa $Sup\mathcal{I}(K_1, K_2)$. ■

Seja agora $\mathcal{IC}_{VU}(K_1, K_2) = \mathcal{I}(K_1, K_2) \cap \mathbb{C}_{VU}(K_1)$ o conjunto de todas as sublinguagens de K_1 que são vu-controláveis e.r.a. L e interconsistentes e.r.a. K_2. Como $\mathcal{I}(K_1, K_2)$ e $\mathbb{C}_{VU}(K_1)$ são não vazios e fechados sob uniões arbitrárias, então $\mathcal{IC}_{VU}(K_1, K_2)$ também o é. Assim, $\mathcal{IC}_{VU}(K_1, K_2)$ contém um (único) elemento supremo, o qual denota-se por $Sup\mathcal{IC}_{VU}(K_1, K_2)$.

Suponha que K_1 e K_2 são vu-controláveis e.r.a. L mas que não são interconsistentes. Em geral, $Sup\mathcal{I}(K_1, K_2)$ pode não ser vu-controlável e.r.a. L, isto é, o cálculo de $Sup\mathcal{I}$ pode destruir a controlabilidade. Suponha agora que K_1 e K_2 são interconsistentes, mas que K_1 não é vu-controlável e.r.a. L. É possível que $Sup\mathbb{C}_{VU}(K_1)$ e K_2 não sejam interconsistentes, isto é, o cálculo de $Sup\mathbb{C}_{VU}$ pode destruir a interconsistência. O algoritmo para obtenção de $Sup\mathcal{IC}_{VU}$ alterna o cálculo de $Sup\mathbb{C}_{VU}$ e $Sup\mathcal{I}$ até que ambas as propriedades (vu-controlabilidade e interconsistência) sejam alcançadas. Pode-se mostrar que este procedimento converge em um número finito de iterações. O exemplo apresentado a seguir ilustra o procedimento para o cálculo de $Sup\mathcal{IC}_{VU}$, mostrando ainda a necessidade deste procedimento ser iterativo.

Exemplo 5.2 *Considere que o autômato mostrado na Figura 5.6 consiste no modelo de uma planta* $H = (L, \Gamma)$.

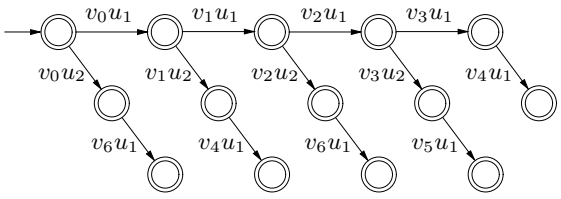
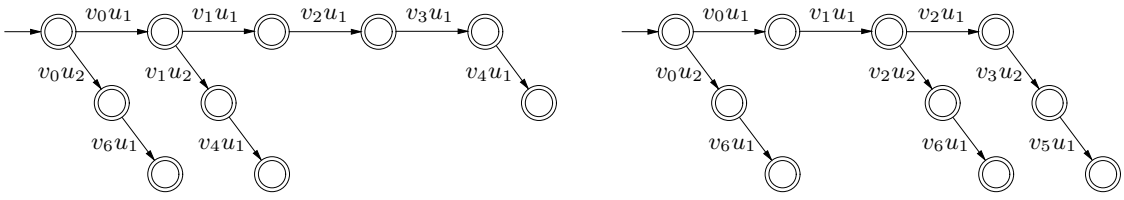


Figura 5.6: Autômato que reconhece a linguagem L.

Sejam K_1 e K_2 duas linguagens reconhecidas pelos autômatos mostrados na Figura 5.7. Pode-se facilmente verificar que estas duas linguagens são vu-controláveis e.r.a. L.

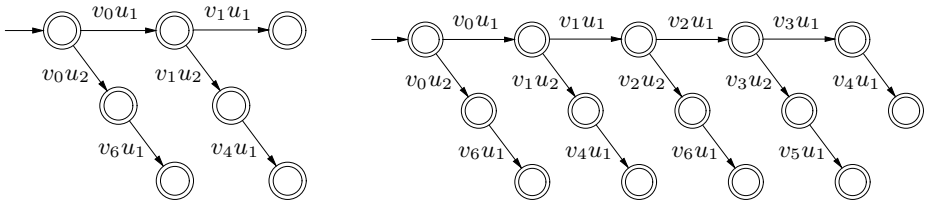


(a) Autômato para K_1 (b) Autômato para K_2

Figura 5.7: Linguagens K_1 e K_2 para ilustrar o cálculo de $Sup\mathcal{IC}_{VU}(K_1, K_2)$.

Observe, entretanto, que K_1 e K_2 não são interconsistentes. Note que após a cadeia $w = v_0u_1 \circ v_1u_1 \circ v_2u_1 \in \overline{K_1} \cap \overline{K_2}$ tem-se que $V_{K_1}(w) \cap V_{K_2}(w) = \{v_3\}$, mas que $V_{K_1 \cap K_2}(w) = \emptyset$.

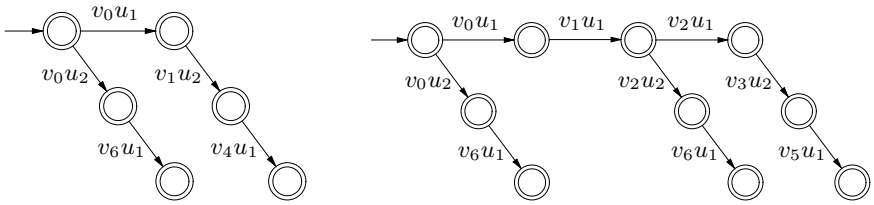
Calcula-se então a máxima sublinguagem de K_1 que é interconsistente em relação a K_2, denotada por $Sup\mathcal{I}(K_1, K_2)$, chamada de K_1' neste exemplo. Um autômato que reconhece esta linguagem é mostrado na Figura 5.8(a).



(a) Autômato para K_1' (b) Modelo para a planta H

Figura 5.8: Autômatos que reconhecem as linguagens K_1' e L.

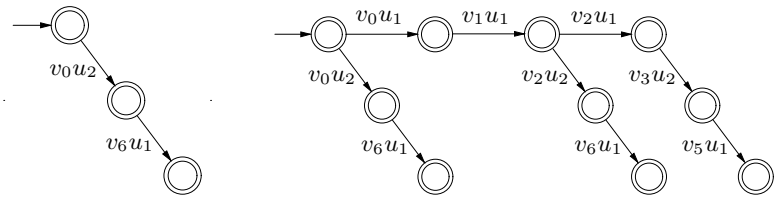
Note agora que a linguagem reconhecida pelo autômato obtido não é vu-controlável e.r.a. L, pois para a cadeia $s = v_0u_1 \circ v_1u_1 \in \overline{K_1'}$ existe um evento (v_2) que pode ocorrer na planta e que não é contemplado em K_1', ou, formalmente, $\exists s \in \overline{K_1'} : V_L(s) \nsubseteq V_{K_1'}(s)$. Sendo assim, calcula-se $K_1'' = Sup\mathbb{C}_{VU}(K_1')$. O autômato que reconhece K_1'' é mostrado na Figura 5.9(a).



(a) Autômato para K_1'' (b) Autômato para K_2

Figura 5.9: Autômatos que reconhecem as linguagens K_1'' e K_2.

Mais uma vez, tem-se que K_1'' e K_2 não são interconsistentes e então obtém-se $K_1''' = Sup\mathcal{I}(K_1'', K_2)$ (ver Figura 5.10(a)). Finalmente, verifica-se que K_1''' é vu-controlável e.r.a. L e então $K_1''' = Sup\mathcal{I}\mathbb{C}_{VU}(K_1, K_2)$, ou seja, que K_1''' é a máxima sublinguagem de K_1 que é interconsistente em relação a K_2 e também é vu-controlável e.r.a. L.



(a) Autômato para K_1''' (b) Autômato para K_2

Figura 5.10: Autômatos que reconhecem as linguagens K_1''' e K_2.

■

A seguir, apresentam-se alguns resultados parciais que serão utilizados para provar o Teorema 5.1 e a Proposição 5.2, os principais resultados deste capítulo.

Lema 5.1 *Sejam $K_1, K_2 \subseteq (V^+ \times U)^*$ duas linguagens tais que $K_1 \subseteq K_2$. Então K_1 e K_2 são interconsistentes.*

Prova: *Sejam K_1 e K_2 tais que $K_1 \subseteq K_2$. Então $K_1 \cap K_2 = K_1$ e assim $\overline{K_1 \cap K_2} = \overline{K_1}$. Além disso, para $K_1 \subseteq K_2$ tem-se que $\overline{K_1} \subseteq \overline{K_2}$, o que implica $\overline{K_1} \cap \overline{K_2} = \overline{K_1}$. Sendo assim, $\overline{K_1} = \overline{K_1} \cap \overline{K_2} = \overline{K_1 \cap K_2}$. Mas se $K_1 \subseteq K_2$, então $(\forall w \in \overline{K_1})\ V_{K_1}(w) \subseteq V_{K_2}(w)$, o que implica $(\forall w \in \overline{K_1})\ V_{K_1}(w) \cap V_{K_2}(w) = V_{K_1}(w)$. Ainda, como $K_1 = K_1 \cap K_2$, então $(\forall w \in \overline{K_1})\ V_{K_1}(w) = V_{K_1 \cap K_2}(w)$. Logo, pode-se afirmar que $(\forall w \in \overline{K_1})\ V_{K_1}(w) \cap V_{K_2}(w) = V_{K_1 \cap K_2}(w)$. Por fim, como $\overline{K_1} = \overline{K_1} \cap \overline{K_2}$, então tem-se que $(\forall w \in \overline{K_1} \cap \overline{K_2})\ V_{K_1}(w) \cap V_{K_2}(w) = V_{K_1 \cap K_2}(w)$, ou seja, K_1 e K_2 são interconsistentes.* ◇

Proposição 5.2 *Seja $L \subseteq (V^+ \times U)^*$ e $E_1, E_2 \subseteq P_V(L)$. Seja ainda $K_i{}^{\uparrow} = Sup\mathbb{C}_{VU}[P_V^{-1}(E_i) \cap L]$, onde $i = 1, 2$. Então, $Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow} = Sup\mathcal{IC}_{VU}(K_2{}^{\uparrow}, K_1{}^{\uparrow}) \cap K_1{}^{\uparrow} = (K_1 \cap K_2)^{\uparrow}$.*

Prova: *Sejam $i, j = 1, 2$, com $i \neq j$.*

a) $Sup\mathcal{IC}_{VU}(K_i{}^{\uparrow}, K_j{}^{\uparrow}) \cap K_j{}^{\uparrow} \subseteq (K_i \cap K_j)^{\uparrow}$.

Sabe-se que $Sup\mathcal{IC}_{VU}(K_i{}^{\uparrow}, K_j{}^{\uparrow}) \subseteq K_i$ e que $K_j{}^{\uparrow} \subseteq K_j$, logo $Sup\mathcal{IC}_{VU}(K_i{}^{\uparrow}, K_j{}^{\uparrow}) \cap K_j{}^{\uparrow} \subseteq K_i \cap K_j$. Agora, como $Sup\mathcal{IC}_{VU}(K_i{}^{\uparrow}, K_j{}^{\uparrow})$ e $K_j{}^{\uparrow}$ são vu-controláveis e interconsistentes, então pela Proposição 4.2 sabe-se que $Sup\mathcal{IC}_{VU}(K_i{}^{\uparrow}, K_j{}^{\uparrow}) \cap K_j{}^{\uparrow}$ é vu-controlável e.r.a. L. Desta forma, $Sup\mathcal{IC}_{VU}(K_i{}^{\uparrow}, K_j{}^{\uparrow}) \cap K_j{}^{\uparrow} \subseteq (K_i \cap K_j)^{\uparrow}$.

b) $Sup\mathcal{IC}_{VU}(K_i{}^{\uparrow}, K_j{}^{\uparrow}) \cap K_j{}^{\uparrow} \supseteq (K_i \cap K_j)^{\uparrow}$.

Sabe-se que $K_i \cap K_j \subseteq K_j$, o que implica $(K_i \cap K_j)^{\uparrow} \subseteq K_j{}^{\uparrow}$. Assim, pelo Lema 5.1 conclui-se que $(K_i \cap K_j)^{\uparrow}$ e $K_j{}^{\uparrow}$ são interconsistentes. Agora, como $(K_i \cap K_j)^{\uparrow} \subseteq K_i{}^{\uparrow}$ e considerando que $(K_i \cap K_j)^{\uparrow}$ e $K_j{}^{\uparrow}$ são interconsistentes, tem-se que $(K_i \cap K_j)^{\uparrow} \in \mathcal{IC}_{VU}(K_i{}^{\uparrow}, K_j{}^{\uparrow})$ e que assim $(K_i \cap K_j)^{\uparrow} \subseteq Sup\mathcal{IC}_{VU}(K_i{}^{\uparrow}, K_j{}^{\uparrow})$. Mas se $(K_i \cap K_j)^{\uparrow} \subseteq K_j{}^{\uparrow}$ e $(K_i \cap K_j)^{\uparrow} \subseteq Sup\mathcal{IC}_{VU}(K_i{}^{\uparrow}, K_j{}^{\uparrow})$, então $(K_i \cap K_j)^{\uparrow} \subseteq Sup\mathcal{IC}_{VU}(K_i{}^{\uparrow}, K_j{}^{\uparrow}) \cap K_j{}^{\uparrow}$.

Sendo assim, $Sup\mathcal{IC}_{VU}(K_i{}^{\uparrow}, K_j{}^{\uparrow}) \cap K_j{}^{\uparrow} = (K_i \cap K_j)^{\uparrow}$.

Fazendo $i = 1$ e $j = 2$ obtém-se $Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow} = (K_1 \cap K_2)^{\uparrow}$. Já para $i = 2$ e $j = 1$ tem-se que $Sup\mathcal{IC}_{VU}(K_2{}^{\uparrow}, K_1{}^{\uparrow}) \cap K_1{}^{\uparrow} = (K_2 \cap K_1)^{\uparrow} = (K_1 \cap K_2)^{\uparrow} = Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow}$. Logo, $Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow} = Sup\mathcal{IC}_{VU}(K_2{}^{\uparrow}, K_1{}^{\uparrow}) \cap K_1{}^{\uparrow}$.

◇

Teorema 5.1 *Sejam uma planta* $H=(L,\Gamma)$ *e duas especificações* $E_1, E_2 \subseteq P_V(L)$. *Seja ainda* $K_i{}^{\uparrow}=Sup\mathbb{C}_{VU}[P_V^{-1}(E_i)\cap L]$, *onde* $i=1,2$. *As seguintes afirmações são equivalentes:*

1. *Existem dois SMN,* F_1, F_2, *tais que* $F_1 \wedge F_2$ *é um SMN e.r.a.* $E_1 \cap E_2$ *e que* $P_V[L_m((F_1 \wedge F_2)/H)] = Sup\mathbb{C}_V(E_1 \cap E_2) \neq \emptyset$;
2. $Sup\mathcal{I}\mathbb{C}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow})$ *e* $K_2{}^{\uparrow}$ *são não vazias;*
3. $Sup\mathcal{I}\mathbb{C}_{VU}(K_2{}^{\uparrow}, K_1{}^{\uparrow})$ *e* $K_1{}^{\uparrow}$ *são não vazias.*

Prova: *Mostra-se a equivalência entre (2) e (3) e também a equivalência entre (1) e (2), o que garante a equivalência entre (1) e (3).*

a) Equivalência entre (2) e (3).

Sejam $Sup\mathcal{I}\mathbb{C}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow})$ *e* $K_2{}^{\uparrow}$ *não vazias. Então, de acordo com a Proposição 4.4 tem-se que* $Sup\mathcal{I}\mathbb{C}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow} \neq \emptyset$. *Mas pela Proposição 5.2 sabe-se que* $Sup\mathcal{I}\mathbb{C}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow} = Sup\mathcal{I}\mathbb{C}_{VU}(K_2{}^{\uparrow}, K_1{}^{\uparrow}) \cap K_1{}^{\uparrow}$ *e portanto tem-se que* $Sup\mathcal{I}\mathbb{C}_{VU}(K_2{}^{\uparrow}, K_1{}^{\uparrow}) \cap K_1{}^{\uparrow} \neq \emptyset$. *Logo,* $Sup\mathcal{I}\mathbb{C}_{VU}(K_2{}^{\uparrow}, K_1{}^{\uparrow})$ *e* $K_1{}^{\uparrow}$ *são não vazias. Da mesma forma, se* $Sup\mathcal{I}\mathbb{C}_{VU}(K_2{}^{\uparrow}, K_1{}^{\uparrow})$ *e* $K_1{}^{\uparrow}$ *são não vazias, então* $Sup\mathcal{I}\mathbb{C}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow})$ *e* $K_2{}^{\uparrow}$ *também são não vazias.*

b) Equivalência entre (1) e (2).

(2 ⇒ 1) *Supor que* $Sup\mathcal{I}\mathbb{C}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow})$ *e* $K_2{}^{\uparrow}$ *são não vazias, onde* $K_i{}^{\uparrow} = Sup\mathbb{C}_{VU}[P_V^{-1}(E_i) \cap L]$. *Tendo em vista que* $Sup\mathcal{I}\mathbb{C}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow})$ *e* $K_2{}^{\uparrow}$ *são linguagens vu-controláveis e.r.a.* L, *pela Proposição 3.2 pode-se afirmar que existem SMN* F_1, F_2 *que implementem estas linguagens. Sejam então* F_1, F_2 *SMN tais que* $L_m(F_1/H) = Sup\mathcal{I}\mathbb{C}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow})$ *e* $L_m(F_2/H) = K_2{}^{\uparrow}$. *Sabe-se, por definição, que* $Sup\mathcal{I}\mathbb{C}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow})$ *e* $K_2{}^{\uparrow}$ *são interconsistentes e assim, de acordo com a Proposição 4.3, tem-se que* $F_1 \wedge F_2$ *é um supervisor consistente para* H. *Ainda pela Proposição 4.3 tem-se que* $L_m((F_1 \wedge F_2)/H) = L_m(F_1/H) \cap L_m(F_2/H)$. *Assim,* $\overline{L_m((F_1 \wedge F_2)/H)} = \overline{L_m(F_1/H) \cap L_m(F_2/H)} = \overline{Sup\mathcal{I}\mathbb{C}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow}}$. *Mas como* $Sup\mathcal{I}\mathbb{C}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow})$ *e* $K_2{}^{\uparrow}$ *são interconsistentes, então (Proposição 4.1)* $\overline{Sup\mathcal{I}\mathbb{C}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow}} = \overline{Sup\mathcal{I}\mathbb{C}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow})} \cap \overline{K_2{}^{\uparrow}}$. *Desta forma,* $\overline{L_m((F_1 \wedge F_2)/H)} = \overline{Sup\mathcal{I}\mathbb{C}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow})} \cap \overline{K_2{}^{\uparrow}} = \overline{L_m(F_1/H)} \cap \overline{L_m(F_2/H)}$. *Agora, como os supervisores* F_i *são SMN e.r.a.* E_i, *então* $\overline{L_m(F_i/H)} = L(F_i/H)$, *onde* $L_m(F_i/H) = L(F_i/H) \cap P_V^{-1}(E_i)$. *Portanto,*

$\overline{L_m((F_1 \wedge F_2)/H)} = L(F_1/H) \cap L(F_2/H) = L((F_1 \wedge F_2)/H)$. *Sendo assim, pode-se afirmar que* $F_1 \wedge F_2$ *é um SMN e.r.a.* $E_1 \cap E_2$.

Sabe-se que $L_m((F_1 \wedge F_2)/H) = L_m(F_1/H) \cap L_m(F_2/H) = Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow}$. *Agora, uma vez que* $Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow})$ *e* $K_2{}^{\uparrow}$ *são não vazias e interconsistentes, então pela Proposição 4.4 sabe-se que* $Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow} \neq \emptyset$. *Mas, conforme mostrado na prova da Proposição 5.2,* $Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow} = (K_1 \cap K_2)^{\uparrow}$ *e portanto* $L_m((F_1 \wedge F_2)/H) = (K_1 \cap K_2)^{\uparrow} \neq \emptyset$. *Ainda, pela Proposição 3.7 sabe-se que* $(K_1 \cap K_2)^{\uparrow} \neq \emptyset \Rightarrow P_V[(K_1 \cap K_2)^{\uparrow}] \neq \emptyset$, *de onde se conclui que* $P_V[L_m((F_1 \wedge F_2)/H)] = P_V[Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow}] = P_V[(K_1 \cap K_2)^{\uparrow}] \neq \emptyset$.

Resta mostrar então que $P_V[(K_1 \cap K_2)^{\uparrow}] = Sup\mathbb{C}_V(E_1 \cap E_2)$.

Sabe-se que $K_1 \cap K_2 = [P_V^{-1}(E_1) \cap L] \cap [P_V^{-1}(E_2) \cap L] = [P_V^{-1}(E_1) \cap P_V^{-1}(E_2)] \cap L$. *Mas* $P_V^{-1}(E_1) \cap P_V^{-1}(E_2) = P_V^{-1}(E_1 \cap E_2)$ *(Lema 4.1) e assim,* $K_1 \cap K_2 = P_V^{-1}(E_1 \cap E_2) \cap L$. *Agora, como* $E_1 \cap E_2 \subseteq P_V(L)$, *então pela Proposição 3.7 pode-se afirmar que* $P_V[(K_1 \cap K_2)^{\uparrow}] = Sup\mathbb{C}_V(E_1 \cap E_2)$.

(1 ⇒ 2) *Conforme já mostrado nesta prova, para* $E_i \subseteq P_V(L)$ *e* $K_i{}^{\uparrow} = Sup\mathbb{C}_{VU}[P_V^{-1}(E_i) \cap L]$, *com* $i = 1, 2$, *tem-se que* $P_V[Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow}] = P_V[(K_1 \cap K_2)^{\uparrow}] = Sup\mathbb{C}_V(E_1 \cap E_2)$. *Supor então que* $Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) = \emptyset$ *ou* $K_2{}^{\uparrow} = \emptyset$. *Assim,* $Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow} = \emptyset$ *e portanto* $P_V[Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow}] = \emptyset$. *Desta forma, conclui-se que* $Sup\mathbb{C}_V(E_1 \cap E_2) = \emptyset$, *ou seja, que não existem SMN* F_1 *e* F_2 *tais que* $P_V[L_m((F_1 \wedge F_2)/H)] = Sup\mathbb{C}_V(E_1 \cap E_2) \neq \emptyset$.

◇

Sejam então F_i e F_j dois supervisores tais que $L_m(F_i/H) = Sup\mathcal{IC}_{VU}(K_i{}^{\uparrow}, K_j{}^{\uparrow}) \neq \emptyset$ e $L_m(F_j/H) = K_j{}^{\uparrow} \neq \emptyset$, onde $i, j = 1, 2$ com $i \neq j$. Pelo Teorema 5.1 pode-se concluir que $F_1 \wedge F_2$ é um SMN e.r.a. $E_1 \cap E_2$ e que $P_V[L_m((F_1 \wedge F_2)/H)] = Sup\mathbb{C}_V(E_1 \cap E_2) \neq \emptyset$.

Observe que se as linguagens $K_i{}^{\uparrow}$ e $K_j{}^{\uparrow}$ não são interconsistentes, então o supervisor F_i obtido de forma que $L_m(F_i/H) = Sup\mathcal{IC}_{VU}(K_i{}^{\uparrow}, K_j{}^{\uparrow})$ não implementa a máxima linguagem v-controlável contida na especificação E_i, ou seja, $P_V[L_m(F_i/H)] \subsetneqq Sup\mathbb{C}_V(E_i)$. Por outro lado, o procedimento apresentado neste trabalho garante que a ação conjunta dos supervisores é tal que $P_V[L_m((F_1 \wedge F_2)/H)] = Sup\mathbb{C}_V(E_1 \cap E_2) \neq \emptyset$, isto é, mesmo não sendo minimamente restritivo para uma especificação particular, o supervisor conjunção é não bloqueante e minimamente restritivo para a especificação global.

Sendo assim, o Teorema 5.1 indica que se na síntese modular de supervisores encontram-se duas linguagens ${K_1}^{\uparrow}$ e ${K_2}^{\uparrow}$ que não são interconsistentes, então deve-se fixar uma destas linguagens e calcular a máxima sublinguagem da outra linguagem que é interconsistente e.r.a. linguagem fixada e que é *vu*-controlável e.r.a. L. É importante observar ainda que a otimalidade da solução é garantida independentemente da linguagem (${K_1}^{\uparrow}$ ou ${K_2}^{\uparrow}$) que é fixada.

Com base nos resultados anteriormente obtidos, pode-se estabelecer o seguinte procedimento para a resolução de um problema de síntese de supervisores modulares para sistemas híbridos [1].

1. Seja $H = (L, \Gamma)$ o modelo SED para uma planta híbrida $\mathcal{H}$. Dadas duas especificações $E_1, E_2 \subseteq V^*$, obter as especificações equivalentes $K_1, K_2 \subseteq L \subseteq (V^+ \times U)^*$ fazendo $K_i = P_V^{-1}(E_i) \cap L$, onde $i = 1, 2$;

2. Verificar se cada linguagem K_i é *vu*-controlável em relação a L. Se for, chamar $K_i = {K_i}^{\uparrow}$ e pular para o passo 3. Caso contrário, deve-se obter a máxima linguagem *vu*-controlável contida em K_i (apenas para K_i não controlável), denotada por ${K_i}^{\uparrow}$;

3. Testar se ${K_1}^{\uparrow}$ e ${K_2}^{\uparrow}$ são não vazias e interconsistentes. Se forem, então os supervisores F_i são implementados através de autômatos tais que $L_m(F_i/H) = {K_i}^{\uparrow}$. Assim, de acordo com o Teorema 4.1, a ação conjunta dos supervisores modulares será não bloqueante e minimamente restritiva. Se as linguagens são não vazias mas não são interconsistentes, pular para o passo 4;

4. Calcular $Sup\mathcal{IC}_{_{VU}}({K_1}^{\uparrow}, {K_2}^{\uparrow})$. Se esta linguagem é não vazia, então, de acordo com o Teorema 5.1, pode-se afirmar que os supervisores F_1, F_2 tais que $L_m(F_1/H) = Sup\mathcal{IC}_{_{VU}}({K_1}^{\uparrow}, {K_2}^{\uparrow})$ e $L_m(F_2/H) = {K_2}^{\uparrow}$ levam a uma ação conjunta que é não bloqueante e minimamente restritiva. Obtém-se então $L_m((F_1 \wedge F_2)/H) = Sup\mathcal{IC}_{_{VU}}({K_1}^{\uparrow}, {K_2}^{\uparrow}) \cap {K_2}^{\uparrow}$. Pode-se determinar ainda a máxima linguagem v-controlável contida em $E_1 \cap E_2$, fazendo $Sup\mathbb{C}_{_V}(E_1 \cap E_2) = P_V[Sup\mathcal{IC}_{_{VU}}({K_1}^{\uparrow}, {K_2}^{\uparrow}) \cap {K_2}^{\uparrow}]$.

Obs.: Conforme mostrado no Teorema 5.1, o mesmo resultado é encontrado se no passo 4 calcula-se $Sup\mathcal{IC}_{_{VU}}({K_2}^{\uparrow}, {K_1}^{\uparrow})$ e os supervisores são tais que $L_m(F_1/H) = {K_1}^{\uparrow}$ e $L_m(F_2/H) = Sup\mathcal{IC}_{_{VU}}({K_2}^{\uparrow}, {K_1}^{\uparrow})$. Neste caso, a máxima linguagem v-controlável contida em $E_1 \cap E_2$ é obtida fazendo $Sup\mathbb{C}_{_V}(E_1 \cap E_2) = P_V[Sup\mathcal{IC}_{_{VU}}({K_2}^{\uparrow}, {K_1}^{\uparrow}) \cap {K_1}^{\uparrow}]$.

[1] Considera-se conhecido o modelo lógico para a planta híbrida

Conforme explicado no Capítulo 3, o projeto individual dos supervisores modulares pode ser feito com auxílio do GRAIL. Além disso, o teste da interconsistência e o cálculo de $Sup\mathcal{IC}_{VU}$ também podem ser feitos com auxílio do GRAIL. Estas duas últimas funções foram desenvolvidas no contexto do trabalho (Rodrigues, 2004).

O exemplo apresentado na seqüência ilustra o procedimento de síntese de supervisores modulares para sistemas híbridos. Neste exemplo, as linguagens inicialmente encontradas para os supervisores não são interconsistentes e então utilizam-se os resultados obtidos nesta seção de forma a garantir que os supervisores modulares sejam não conflitantes.

Exemplo 5.3 *Considere que o autômato mostrado na Figura 5.11 consiste no modelo SED $H = (L, \Gamma)$ para um sistema híbrido $\mathcal{H}$.*

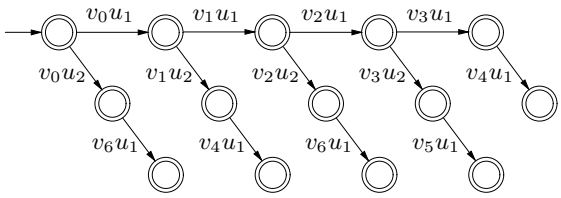
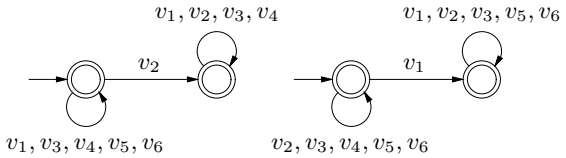


Figura 5.11: Autômato que reconhece a linguagem L.

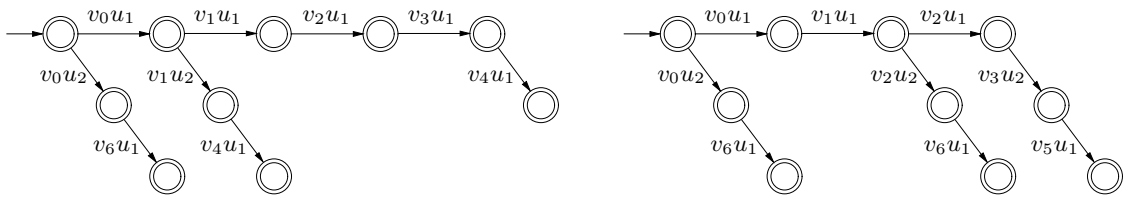
Sejam $E_1, E_2 \subseteq V^*$ as especificações para o comportamento deste sistema em malha fechada (ver Figura 5.12). A especificação E_1 requer que, uma vez tendo ocorrido o evento v_2, os eventos v_5 e v_6 sejam proibidos de acontecer. De forma semelhante, a especificação E_2 proíbe a ocorrência de v_4 a partir da ocorrência do evento v_1. Deseja-se sintetizar supervisores modulares para atender estas especificações.



(a) $E_1 \subseteq V^*$ (b) $E_2 \subseteq V^*$

Figura 5.12: Especificações $E_1, E_2 \subseteq V^*$.

Ao final dos dois primeiros passos da síntese dos supervisores modulares, obtém-se ${K_1}^{\uparrow}$ e ${K_2}^{\uparrow}$, onde ${K_i}^{\uparrow} = Sup\mathbb{C}_{VU}[P_V^{-1}(E_i) \cap L]$. Os autômatos mostrados na figura 5.13 reconhecem estas linguagens.



(a) Autômato para $K_1{}^{\uparrow}$ (b) Autômato para $K_2{}^{\uparrow}$

Figura 5.13: Autômatos que reconhecem as linguagens $K_1{}^{\uparrow}$ e $K_2{}^{\uparrow}$.

No terceiro passo verifica-se que as linguagens obtidas anteriormente são não vazias, mas que não são interconsistentes. Observe que estas mesmas linguagens, bem como o autômato para a planta H, foram utilizados no Exemplo 5.2, onde a mesma constatação é feita. Observe ainda que o cálculo de $Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow})$, a ser efetuado no passo 4, já foi realizado naquele exemplo. O autômato que reconhece esta linguagem é mostrado novamente na Figura 5.14.

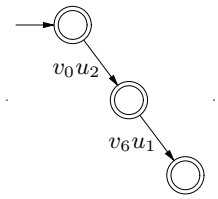


Figura 5.14: Autômato para $Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow})$.

Pode-se verificar que $Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \neq \emptyset$ e então, de acordo com o Teorema 5.1, sabe-se que os supervisores F_1 e F_2 que implementam em malha fechada as linguagens $Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow})$ e $K_2{}^{\uparrow}$, respectivamente, são tais que sua ação concorrente é ótima. Estes supervisores podem ser implementados pelos autômatos mostrados nas figuras 5.14 e 5.13(b), respectivamente. Desta forma, $L_m((F_1 \wedge F_2)/H) = Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow}$. Observe que, neste exemplo, $Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow} = Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) = v_0u_2 \circ v_6u_1$. Assim, $P_V[L_m((F_1 \wedge F_2)/H)] = P_V[Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) \cap K_2{}^{\uparrow}] = v_6$.

De acordo com a Proposição 5.2, o mesmo resultado é obtido fixando-se a linguagem $K_1{}^{\uparrow}$ e calculando $Sup\mathcal{IC}_{VU}(K_2{}^{\uparrow}, K_1{}^{\uparrow})$.

No que segue, sintetiza-se um supervisor monolítico de forma a comparar os resultados obtidos nas duas abordagens. Obtém-se então $K = P_V^{-1}(E_1 \cap E_2) \cap L$, linguagem esta reconhecida pelo autômato mostrado na Figura 5.15.

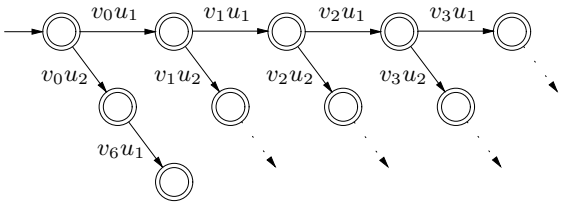


Figura 5.15: Autômato para $K = P_V^{-1}(E_1 \cap E_2) \cap L$.

Pode-se facilmente verificar que o autômato da Figura 5.14 reconhece a linguagem $Sup\mathbb{C}_{VU}(K)$ *e que portanto o autômato mostrado naquela figura pode ser a implementação do supervisor monolítico. Sendo assim,* $L_m(F/H) = v_0u_2 \circ v_6u_1$ *e* $P_V[L_m(F/H)] = v_6$*, ou seja, o resultado obtido através da abordagem monolítica é o mesmo obtido através da abordagem modular, conforme esperado.* ■

Retornando ao problema de síntese modular para sistemas híbridos (MSHS), pode-se estabelecer o seguinte resultado.

Corolário 5.1 *Seja* H *o modelo de sistemas a eventos discretos para a planta híbrida* $\mathcal{H}$ *e sejam* F_i $(i = 1, 2)$ *supervisores SED obtidos como indicado no Teorema 5.1. Os supervisores condição/evento* $\mathcal{F}_i$ *equivalentes aos supervisores* F_i *levam a uma solução ótima para o problema de síntese de supervisores modulares para sistemas híbridos.*

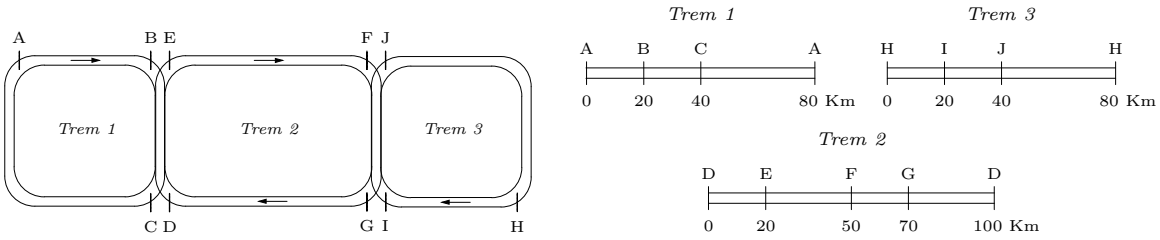
Prova: *Segue diretamente da Proposição 4.6.* ◇

Note que o Corolário 5.1 indica um procedimento para a solução do problema MSHS de forma a garantir a otimalidade de sua solução. Entretanto, em geral não se obtêm os supervisores condição/evento, mas utilizam-se os supervisores SED como implementação discreta dos mesmos.

5.3 Exemplo

No intuito de ilustrar a abordagem proposta, utiliza-se nesta seção o exemplo do sistema de três trens sobre trilhos cíclicos considerado no Capítulo 4 (Seção 4.4). Com exceção das condições iniciais do sistema, todas as demais informações e características foram mantidas.

As condições iniciais (posições dos trens) foram alteradas de forma a explorar os resultados apresentados neste capítulo. A figura que ilustra o sistema em questão é repetida aqui (ver Figura 5.16) a fim de facilitar a visualização do problema.



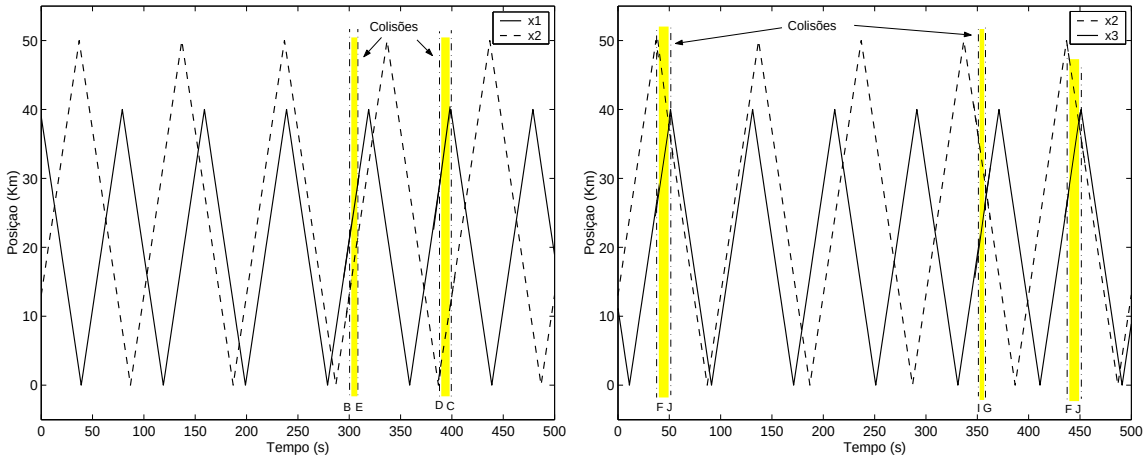
(a) Sistema de Trens (b) Detalhe dos trilhos

Figura 5.16: Três trens sobre trilhos cíclicos.

As novas posições iniciais para os trens são as seguintes. O trem 1 inicialmente está localizado 41 km após a posição A (trecho $\overline{CA}$), o trem 2 está 13 km após D (trecho $\overline{DE}$), e o trem 3 inicia sua viagem a partir do ponto localizado 69 km depois de H (trecho $\overline{JH}$).

Com esta modificação o modelo obtido é diferente do anterior. Entretanto, o número de estados e de transições do novo modelo é igual ao do modelo anterior, ou seja, o autômato de estados finitos que representa o comportamento lógico para este novo sistema também possui 238 estados e 264 transições.

Através da simulação do comportamento do sistema em malha aberta, na qual o modo *rápido* de velocidade é sempre escolhido, observa-se a existência de colisões entre os trens nos dois trechos compartilhados (ver Figura 5.17). No intuito de evitar colisões entre os trens,



(a) Trens 1 e 2 (b) Trens 2 e 3

Figura 5.17: Comportamento do sistema em malha aberta.

propõe-se a utilização de dois supervisores modulares, um para evitar colisões entre os trens 1 e 2 (especificação E_1) e outro para evitar colisões entre os trens 2 e 3 (especificação E_2). A especificação referente a não parada do trem 2 é obtida marcando-se apenas o estado inicial de E_1, o que pode ser interpretado como uma especificação para garantir que, após o trem 2 sensibilizar o sensor E, uma nova ocorrência do evento D sempre poderá acontecer.

Com auxílio das funções de controle supervisório desenvolvidas para o GRAIL, obteve-se as linguagens ${K_1}^{\uparrow}$ e ${K_2}^{\uparrow}$, onde ${K_i}^{\uparrow} = Sup\mathbb{C}_{_{VU}}[P_V^{-1}(E_i) \cap L]$, sendo E_1, E_2 especificações de controle. Verificou-se então que estas linguagens *não são interconsistentes* e portanto não levam a uma solução ótima para o problema. Seguindo o procedimento de síntese de supervisores modulares indicado anteriormente neste capítulo, optou-se por fixar ${K_2}^{\uparrow}$ e calcular $Sup\mathcal{I}\mathbb{C}_{_{VU}}({K_1}^{\uparrow}, {K_2}^{\uparrow})$. Este cálculo também é feito com auxílio das funções desenvolvidas para o GRAIL, utilizando iterativamente as funções para o cálculo de $Sup\mathcal{I}$ e de $Sup\mathbb{C}_{_{VU}}$ até que ambas as propriedades (interconsistência e *vu*-controlabilidade) sejam garantidas. O supervisor F_1 encontrado de forma tal que $L_m(F_1/H) = Sup\mathcal{I}\mathbb{C}_{_{VU}}({K_1}^{\uparrow}, {K_2}^{\uparrow})$ possui 54 estados e 57 transições, e o supervisor F_2 tal que $L_m(F_2/H) = {K_2}^{\uparrow}$ possui 92 estados e 96 transições. Os mesmos não são mostrados aqui por uma questão de legibilidade. A ação conjunta destes supervisores pode ser representada por um autômato com 35 estados e 35 transições e sua ação de controle é ótima no sentido de ser não bloqueante e a menos restritiva possível. A Figura 5.18 mostra o autômato obtido pela conjunção $F_1 \wedge F_2$, na qual $u_1 = [r_1 \quad r_2 \quad r_3]^{\mathrm{T}}$ e $u_2 = [r_1 \quad d_2 \quad r_3]^{\mathrm{T}}$, onde r_i refere-se ao modo *rápido* d_i ao modo *devagar* de velocidade do trem i. Note que novamente foi realizada uma agregação de estados de forma a mostrar apenas os eventos associados ao trem 2 (eventos D, E, F, G), uma vez que este é o único que pode ter sua velocidade modificada.

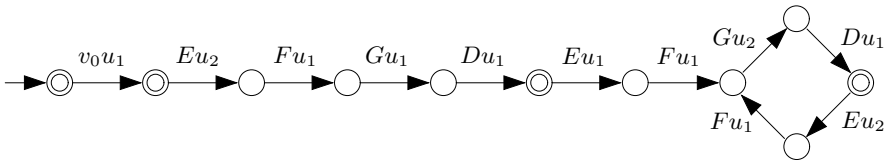
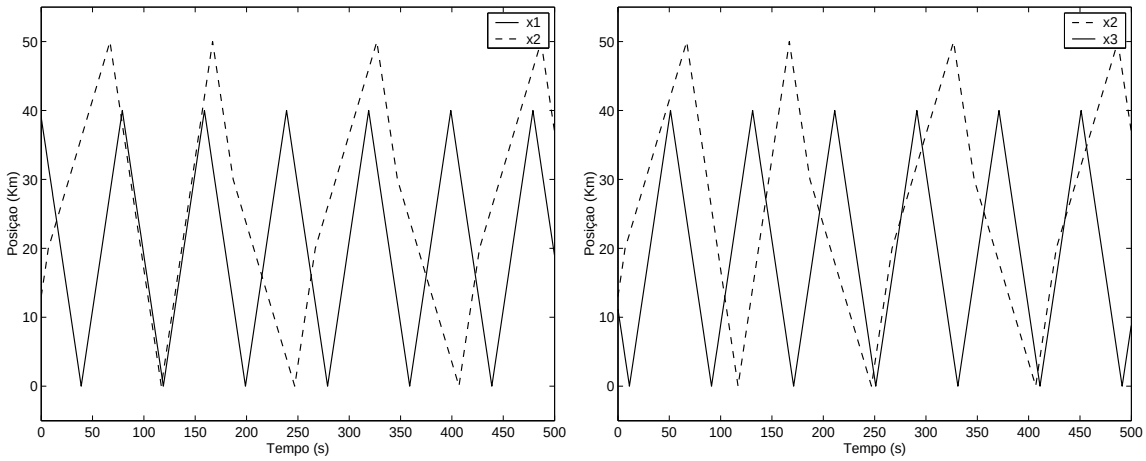


Figura 5.18: Supervisor agregado $F_1 \wedge F_2$.

A Figura 5.19 apresentada a seguir, mostra o resultado de uma simulação para o comportamento da planta sob a ação conjunta dos supervisores modulares. Pode-se verificar que nesta simulação não ocorrem colisões entre os trens, conforme era de se esperar.



(a) Trens 1 e 2 (b) Trens 2 e 3

Figura 5.19: Comportamento do sistema sob supervisão modular.

Por fim, de forma a solucionar o problema original para o sistema híbrido, os supervisores SED encontrados podem ser usados como implementações discretas dos supervisores C/E.

5.4 Conclusões

Conforme resultados estabelecidos no Capítulo 4, para que se tenha a garantia de que a ação conjunta de supervisores modulares seja não bloqueante e minimamente restritiva, as linguagens implementadas em malha fechada pelos supervisores devem ser interconsistentes. Neste capítulo, apresentou-se um procedimento para a síntese de supervisores modulares que sempre garante esta característica, ou seja, sempre leva a uma solução ótima. Desta forma, se existe uma solução para um determinado problema de controle supervisório para sistemas híbridos através da abordagem monolítica, então o mesmo resultado certamente será obtido através da abordagem modular.

Um exemplo foi usado para ilustrar a metodologia proposta, sendo que todo processo de síntese dos supervisores modulares foi feito com auxílio da ferramenta Grail, tendo-se utilizado funções desenvolvidas especificamente para este propósito.

Não se discutiu, neste capítulo, sobre os problemas que podem surgir quando do uso de aproximações para o sistema híbrido. Este tema será objeto de estudo no capítulo subseqüente.

Capítulo 6

Uso de Aproximações para o Sistema Híbrido

6.1 Introdução

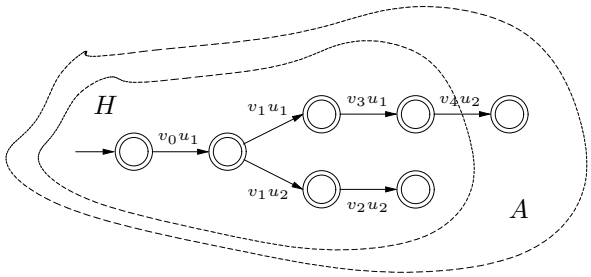
Os resultados estabelecidos nos capítulos anteriores consideram o conhecimento de $H = (L, \Gamma)$, um modelo de eventos discretos cuja linguagem representa o comportamento lógico exato da planta híbrida $\mathcal{H}$.

Mas segundo Cury et al. (1998), pode ser extremamente difícil, ou até mesmo impossível, obter um modelo que represente o comportamento seqüencial exato da planta híbrida e que possua um número finito de estados. Sendo assim, uma alternativa seria a obtenção de um modelo com um número de estados possivelmente infinito, mas cujo comportamento lógico seria exatamente aquele apresentado por $\mathcal{H}$. Entretanto, em geral, a convergência de algoritmos envolvendo modelos de estado é garantida somente para espaços de estados finitos.

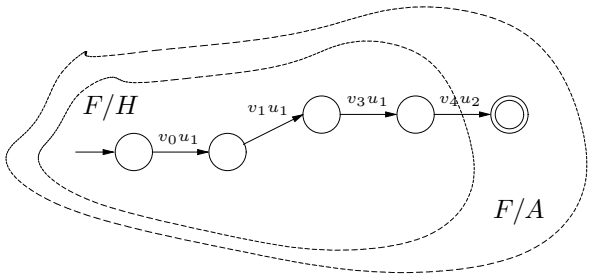
No intuito de contornar este problema, algumas abordagens propõem o uso de aproximações conservadoras para o comportamento do sistema híbrido de forma a solucionar problemas de verificação (Chutinan e Krogh, 2001). No contexto de síntese de supervisores, Cury et al. (1998) propõem a obtenção de um autômato de estados finitos A tal que $L \subset L(A)$, ou seja, A consiste em uma *aproximação externa* do comportamento lógico de $\mathcal{H}$. O autômato A é chamado de *autômato aproximação* para H.

6.2 Problemas Decorrentes do Uso de Aproximações

Seja F um supervisor obtido com base em uma aproximação A de forma a atender uma especificação E que não é prefixo-fechada. Os resultados discutidos anteriormente não garantem que o supervisor F seja não bloqueante para H mesmo quando ele é não bloqueante para A. Sejam, por exemplo, o modelo H e a aproximação A mostrados na Figura 6.1(a). Observe que um supervisor F que implemente a linguagem $L_m(F/A) = v_0u_1 \circ v_1u_1 \circ v_3u_1 \circ v_4u_2$ é não bloqueante para A mas que, no entanto, é bloqueante para H.



(a) Autômatos para H e para A.



(b) Autômatos para F/H e F/A.

Figura 6.1: Modelos para ilustrar o problema de bloqueio.

Alguns trabalhos (p.ex. (Cury et al., 1998)) abordaram este tema e estabeleceram condições sob as quais a propriedade de não bloqueio é preservada para o modelo exato. Entretanto, tais condições recaem sobre aspectos da planta real, os quais não se tem como verificar. Em geral, estudos neste sentido são feitos no intuito de apenas orientar os procedimentos de obtenção e de refinamento de aproximações.

De um modo geral, quando não se conhece o modelo que descreve o comportamento lógico exato para uma planta, não se pode garantir que o supervisor obtido será não bloqueante para esta planta.

No que segue, o estudo dos problemas que surgem na utilização de aproximações para o sistema híbrido é feito apenas para especificações prefixo-fechadas.

6.3 Especificações Prefixo-fechadas

A classe de aproximações introduzida a seguir é muito importante para a continuidade do trabalho

Definição 6.1 *Seja $H=(L,\Gamma)$ o modelo SED para $\mathcal{H}$. Um autômato de estados finitos A é dito ser uma aproximação v-coerente para H se $L \subseteq L(A)$ e*

$$(\forall w \in L)(\forall v \in V_L(w)), U_{L(A)}(w,v) = U_L(w,v). \tag{6.1}$$

Na definição anterior, a condição dada por $L \subseteq L(A)$ significa que A tem que ser uma *aproximação externa* (Cury et al., 1998) para o modelo lógico da planta híbrida. Já a equação (6.1) requer que para toda cadeia w pertencente a linguagem L e para todo evento v possível de ocorrer na planta após w, as condições que podem ser escolhidas em A são as mesmas que podem ser escolhidas em H. Assim, as opções de controle que se tem no modelo aproximado A são as mesmas opções disponíveis na planta real.

A equação (6.1) também pode ser escrita como

$$(\forall w \in L)(\forall v \in V_L(w)), u \in U_{L(A)}(w,v) \Rightarrow w \circ vu \in L$$

A seguir, apresenta-se um exemplo de forma a ilustrar o conceito de v-coerência.

Exemplo 6.1 *Sejam os modelos H e A mostrados na Figura 6.2, onde $H=(L,\Gamma)$ é o autômato que representa o comportamento lógico de uma certa planta híbrida e A um autômato aproximação para H, ou seja, $L \subseteq L(A)$. Observe que para $w = v_0u_1 \circ v_1u_1 \in L$ e para $v_3 \in V_L(w)$ tem-se que $U_L(w,v_3) = \{u_1\}$ enquanto que $U_{L(A)}(w,v_3) = \{u_1,u_2\}$, ou seja, $U_L(w,v_3) \neq U_{L(A)}(w,v_3)$. Sendo assim, A não consiste numa aproximação v-coerente para H.*

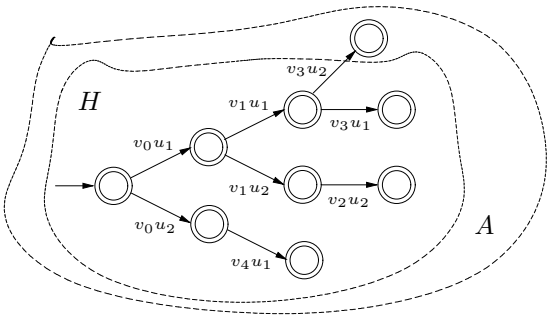


Figura 6.2: Exemplo de aproximação A que não é v-coerente para H.

■

É importante destacar que, em geral, nos algoritmos encontrados na literatura, p.ex. (Chutinan, 1999), as aproximações são obtidas com base nas possibilidades de ocorrência de eventos, e não sobre as escolhas das condições associadas aos eventos. Desta forma, as aproximações obtidas por intermédio destas ferramentas são v-coerentes para o modelo da planta.

6.3.1 Abordagem Monolítica

A seguir apresentam-se alguns resultados que relacionam propriedades de comportamentos e supervisores nos contextos de modelos aproximados e do modelo exato de uma planta híbrida.

Proposição 6.1 *Seja A uma aproximação v-coerente para H. Se uma linguagem prefixo-fechada K é vu-controlável e.r.a. $L(A)$, então $K \cap L$ é vu-controlável e.r.a. L. Além disso, se um supervisor F projetado com base em A é tal que $L_m(F/A) = K$, então $L_m(F/H) = K \cap L$.*

Prova: *Seja $K \subseteq L(A)$ uma linguagem prefixo-fechada. Inicialmente, mostra-se que se $K \cap L$ não é vu-controlável em relação a L, então K não é vu-controlável em relação a $L(A)$.*

Sabe-se que se $K \cap L$ não é vu-controlável em relação a L então $\exists w \in \overline{K \cap L}$ tal que $V_L(w) \nsubseteq V_{K \cap L}(w)$, ou ainda, $\exists w \in \overline{K \cap L}$ para o qual $\exists v \in V_L(w)$ mas $v \notin V_{K \cap L}(w)$. Mas como K e L são prefixo-fechadas, então $\overline{K \cap L} = \overline{K} \cap \overline{L} = K \cap L$, e então pode-se afirmar que $\exists w \in K \cap L$ para o qual $\exists v \in V_L(w)$ mas $v \notin V_{K \cap L}(w)$. Sejam então $w \in K \cap L$ e $v \in V_L(w)$ tais que $v \notin V_{K \cap L}(w)$.

i) Supondo que $v \in V_K(w)$, pode-se afirmar que $\exists u \in U : w \circ vu \in \overline{K}$, ou seja, $u \in U_K(w, v)$. Sabe-se que se $v \in V_K(w)$ e $v \in V_L(w)$, mas $v \notin V_{K \cap L}(w)$, então $\nexists u \in (U_K(w, v) \cap U_L(w, v))$, de onde se pode concluir que $u \in U_K(w, v)$ implica $u \notin U_L(w, v)$. Sabe-se ainda que $K \subseteq L(A)$ e, por isso, $\forall w \in \overline{K}$ e $\forall v \in V_K(w)$, tem-se que $U_K(w, v) \subseteq U_{L(A)}(w, v)$. Logo, pode-se afirmar que $u \in U_{L(A)}(w, v)$ e que, portanto, $(\exists w \in L)(\exists v \in V_L(w))$ tais que $\exists u \in U_{L(A)}(w, v) : u \notin U_L(w, v)$, o que contraria a hipótese de que A é uma aproximação v-coerente para H. Portanto, $v \notin V_K(w)$.

ii) Considerando agora que $v \notin V_K(w)$, tem-se que $\exists w \in K \cap L : \exists v \in V_L(w)$ mas $v \notin V_K(w)$. Sabe-se que $L \subseteq L(A)$ e portanto para $w \in L$ tem-se que $v \in V_L(w) \Rightarrow v \in V_{L(A)}(w)$. Assim, $\exists w \in K \cap L : \exists v \in V_{L(A)}(w)$ mas $v \notin V_K(w)$, ou ainda, $\exists w \in K : V_{L(A)}(w) \nsubseteq V_K(w)$ e portanto K não é vu-controlável em relação a $L(A)$.

No que segue, deve-se mostrar que para um supervisor F tal que $L_m(F/A) = K$, onde K é uma linguagem prefixo-fechada e vu-controlável e.r.a. $L(A)$, então $L_m(F/H) = K \cap L$.

Seja F um supervisor tal que $L_m(F/A) = K$. Como F foi obtido a partir de A e como $L \subseteq L(A)$, então $L_m(F/H) = Sup\mathbb{C}_{VU}(L_m(F/A) \cap L) = Sup\mathbb{C}_{VU}(K \cap L)$. Mas, conforme mostrado anteriormente, para K vu-controlável e.r.a. $L(A)$, tem-se que $K \cap L$ é vu-controlável e.r.a. L e portanto $Sup\mathbb{C}_{VU}(K \cap L) = K \cap L$. Desta forma, $L_m(F/H) = L_m(F/A) \cap L = K \cap L$.

$\diamond$

Sendo assim, se A é uma aproximação v-coerente para H e se K é uma linguagem vu-controlável e.r.a. $L(A)$, então a parte de K que está contida em L é vu-controlável e.r.a. L. De uma forma geral, o resultado apresentado acima estabelece que se A é uma aproximação v-coerente para H, então para qualquer supervisor F projetado com base em A de forma a atender uma especificação prefixo-fechada E, tem-se que $L_m(F/H) = L_m(F/A) \cap L$.

Teorema 6.1 *Seja A uma aproximação v-coerente para uma planta $H = (L, \Gamma)$ e seja um supervisor F projetado para atender uma especificação prefixo-fechada $E \subseteq P_V[L(A)]$ de forma que $P_V[L_m(F/A)] = Sup\mathbb{C}_V(E)$. Então $P_V[L_m(F/H)] \subseteq Sup\mathbb{C}_V(E) \cap P_V(L)$. Além disso, se $P_V[L_m(F/A)] \neq \emptyset$, então $P_V[L_m(F/H)] \neq \emptyset$.*

Prova: *Seja F um supervisor projetado com base em A tal que $P_V[L_m(F/A)] = P_V[Sup\mathbb{C}_{VU}(K)] = Sup\mathbb{C}_V(E)$, onde $K = P_V^{-1}(E) \cap L(A)$. Assim, $L_m(F/A) = Sup\mathbb{C}_{VU}(K)$. Pela Proposição 6.1 pode-se afirmar que se A é uma aproximação v-coerente para H, então $L_m(F/H) = Sup\mathbb{C}_{VU}(K) \cap L$ e portanto $P_V[L(F/H)] = P_V[Sup\mathbb{C}_{VU}(K) \cap L] \subseteq P_V[Sup\mathbb{C}_{VU}(K)] \cap P_V(L)$. Mas $P_V[Sup\mathbb{C}_{VU}(K)] = Sup\mathbb{C}_V(E)$ e então $P_V[L_m(F/H)] \subseteq Sup\mathbb{C}_V(E) \cap P_V(L)$.*

Supor agora que $P_V[L_m(F/A)] \neq \emptyset$. Sabe-se que $L_m(F/H) = L_m(F/A) \cap L$. Como $L_m(F/A)$ e L são linguagens prefixo-fechadas, então $\epsilon \in L_m(F/A) \cap L$. Agora, de acordo com a definição de projeção de linguagens, $P_V(\epsilon) = \epsilon$. Desta forma, $\epsilon \in P_V[L_m(F/A) \cap L] = P_V[L_m(F/H)]$ e portanto $P_V[L_m(F/H)] \neq \emptyset$.

$\diamond$

Por fim, pode-se enunciar o seguinte corolário.

Corolário 6.1 *Dado um problema de Síntese de Supervisores (SSSH) para uma planta híbrida $\mathcal{H}$ e uma especificação prefixo-fechada E. Seja F um supervisor projetado para solucionar este problema na abordagem SED baseado em uma aproximação A que é v-coerente para o modelo lógico exato de $\mathcal{H}$. Então, o supervisor C/E $\mathcal{F}$ equivalente ao supervisor SED F é uma solução para o problema SSSH original.*

Prova: *Segue diretamente do Teorema 6.1 e do Corolário 3.2.*

$\diamond$

Este resultado significa que, um supervisor solução para um dado PCSE, formulado a partir de uma aproximação A que é v-coerente para H, é também uma solução para o PCSE formulado para a planta H. Além disso, o supervisor $\mathcal{F}$ obtido através do Algoritmo 1 apresentado em (González et al., 2001) é uma solução correspondente para o problema original formulado para a planta híbrida $\mathcal{H}$. Entretanto, se não existir um supervisor F tal que $\emptyset \neq P_V[L_m(F/A)] \subseteq E$ então ou a aproximação A é muito grosseira ou a especificação E é muito restritiva. Se a especificação E for muito restritiva, a ponto de não poder ser satisfeita independentemente da precisão da aproximação, então ela precisa ser relaxada para que o problema possa ser solucionado. Assumindo-se que a especificação possa ser satisfeita, resta o caso em que a aproximação é grosseira. Neste caso, Cury e Krogh (1999) indicam um refinamento da aproximação de forma a obter uma aproximação externa mais próxima do comportamento lógico de $\mathcal{H}$. Este assunto será abordado mais à frente neste capítulo.

Conforme explicado anteriormente, existem ferramentas computacionais (p.ex. CheckMate) que possibilitam a obtenção de uma aproximação A que seja v-coerente para H, sendo que esta obtenção é feita sem a necessidade de se conhecer H. Assim, na síntese de supervisores para sistemas híbridos não é necessário obter o modelo $H = (L, \Gamma)$ que representa o comportamento lógico exato para o sistema híbrido $\mathcal{H}$.

6.3.2 Abordagem Modular

No contexto de controle modular de sistemas híbridos, os seguintes resultados podem ser estabelecidos.

Proposição 6.2 *Seja A uma aproximação v-coerente para H e sejam F_1, F_2 dois supervisores SED projetados para A. Se $L_m(F_1/A)$ e $L_m(F_2/A)$ são interconsistentes, então $L_m(F_1/H)$ e $L_m(F_2/H)$ também são interconsistentes.*

Prova: *Supor que* $L_m(F_1/A)$ *e* $L_m(F_2/A)$ *são interconsistentes mas que* $L_m(F_1/H)$ *e* $L_m(F_2/H)$ *não o são. Sabe-se que se* $L_m(F_1/H)$ *e* $L_m(F_2/H)$ *não são interconsistentes, então* $\exists w \in \overline{L_m(F_1/H)} \cap \overline{L_m(F_2/H)}$ *e* $\exists v \in V_{L_m(F_1/H)}(w) \cap V_{L_m(F_2/H)}(w)$ *para os quais* $\nexists u \in U_{L_m(F_1/H)}(w,v) \cap U_{L_m(F_2/H)}(w,v)$. *Pela Proposição 6.1 pode-se afirmar que* $L_m(F_i/H) = L_m(F_i/A) \cap L$ *e que portanto* $L_m(F_i/H) \subseteq L_m(F_i/A)$, *o que implica* $L_m(F_1/H) \cap L_m(F_2/H) \subseteq L_m(F_1/A) \cap L_m(F_2/A)$. *Desta forma,* $w \in \overline{L_m(F_1/H)} \cap \overline{L_m(F_2/H)} \Longrightarrow w \in \overline{L_m(F_1/A)} \cap \overline{L_m(F_2/A)}$. *Além disso, como* $L_m(F_i/H) \subseteq L_m(F_i/A)$, *para* $w \in \overline{L_m(F_i/H)}$ *tem-se que* $V_{L_m(F_i/H)}(w) \subseteq V_{L_m(F_i/A)}(w)$. *Assim, para* $w \in \overline{L_m(F_1/H)} \cap \overline{L_m(F_2/H)}$ *e* $v \in V_{L_m(F_1/H)}(w) \cap V_{L_m(F_2/H)}(w)$ *tem-se que* $w \in \overline{L_m(F_1/A)} \cap \overline{L_m(F_2/A)}$ *e* $v \in V_{L_m(F_1/A)}(w) \cap V_{L_m(F_2/A)}(w)$. *Ainda, como* $L_m(F_i/H) \subseteq L$, *então* $w \in \overline{L_m(F_1/H)} \cap \overline{L_m(F_2/H)} \Longrightarrow w \in L$ *e* $v \in V_{L_m(F_1/H)}(w) \cap V_{L_m(F_2/H)}(w) \Longrightarrow v \in V_L(w)$. *Supor agora que* $L_m(F_1/A)$ *e* $L_m(F_2/A)$ *são interconsistentes. Então, para os mesmos* w *e* v *considerados anteriormente tem-se que* $\exists u \in U_{L_m(F_1/A)}(w,v) \cap U_{L_m(F_2/A)}(w,v)$. *Mas se* $u \in U_{L_m(F_1/A)}(w,v) \cap U_{L_m(F_2/A)}(w,v)$ *então* $w \circ vu \in \overline{L_m(F_1/A)} \cap \overline{L_m(F_2/A)}$, *o que implica* $w \circ vu \in L(A)$ *e* $vu \in F_1(w) \cap F_2(w)$. *Ainda, como* $w \circ vu \in L(A)$ *então* $u \in U_{L(A)}(w,v)$. *Considerando que* A *é uma aproximação* v*-coerente para* H, *pode-se afirmar que para* $\forall w \in L$ *e* $\forall v \in V_L(w)$ *tem-se que* $U_{L(A)}(w,v) = U_L(w,v)$ *e, portanto, para* w *e* v *considerados, tem-se que* $u \in U_{L(A)}(w,v) \Rightarrow u \in U_L(w,v)$ *e então* $w \circ vu \in L$. *Logo,* $w \in \overline{L_m(F_1/H)} \cap \overline{L_m(F_2/H)} \bigwedge w \circ vu \in L \bigwedge vu \in F_1(w) \cap F_2(w)$, *de onde se conclui que* $w \circ vu \in \overline{L_m(F_1/H)} \cap \overline{L_m(F_2/H)}$. *Ou seja, para* w *e* v *considerados* $\exists u \in U_{L_m(F_1/H)}(w,v) \cap U_{L_m(F_2/H)}(w,v)$, *contrariando a hipótese de que* $L_m(F_1/H)$ *e* $L_m(F_2/H)$ *não são interconsistentes. Portanto, se* $L_m(F_1/A)$ *e* $L_m(F_2/A)$ *são interconsistentes, então* $L_m(F_1/H)$ *e* $L_m(F_2/H)$ *são necessariamente interconsistentes.*

$\diamond$

Teorema 6.2 *Seja* A *uma aproximação* v*-coerente para uma planta* $H = (L, \Gamma)$ *e sejam* F_1, F_2 *supervisores projetados para atender duas especificações prefixo-fechadas* $E_1, E_2 \subseteq P_V[L(A)]$ *de forma que* $P_V[L_m((F_1 \wedge F_2)/A)] = Sup\mathbb{C}_V(E_1 \cap E_2)$, *sendo* $L_m(F_1/A)$ *e* $L_m(F_2/A)$ *interconsistentes. Então* $P_V[L_m((F_1 \wedge F_2)/H)] \subseteq Sup\mathbb{C}_V(E_1 \cap E_2) \cap P_V(L)$. *Além disso, se* $P_V[L_m((F_1 \wedge F_2)/A)] \neq \emptyset$, *então* $P_V[L_m((F_1 \wedge F_2)/H)] \neq \emptyset$.

Prova: *Pela Proposição 6.2 sabe-se que para* $L_m(F_1/A)$ *e* $L_m(F_2/A)$ *interconsistentes tem-se que* $L_m(F_1/H)$ *e* $L_m(F_2/H)$ *também são interconsistentes. Agora, pela Proposição 4.3 pode-se afirmar que para* $L_m(F_1/H)$ *e* $L_m(F_2/H)$ *interconsistentes tem-se que* $F_1 \wedge F_2$ *é*

consistente para H. Por definição, sabe-se que $L_m((F_1 \wedge F_2)/H) = L_m(F_1/H) \cap L_m(F_2/H)$, mas $L_m(F_i/H) = L_m(F_i/A) \cap L$ (Proposição 6.1) e então $L_m((F_1 \wedge F_2)/H) = [L_m(F_1/A) \cap L] \cap [L_m(F_2/A) \cap L] = [L_m(F_1/A) \cap L_m(F_2/A)] \cap L = L_m((F_1 \wedge F_2)/A) \cap L$. Sendo assim, $P_V[L_m((F_1 \wedge F_2)/H)] = P_V[L_m((F_1 \wedge F_2)/A) \cap L] \subseteq P_V[L_m((F_1 \wedge F_2)/A)] \cap P_V(L) = Sup\mathbb{C}_V(E_1 \cap E_2) \cap P_V(L)$. Portanto, $P_V[L_m((F_1 \wedge F_2)/H)] \subseteq Sup\mathbb{C}_V(E_1 \cap E_2) \cap P_V(L)$.

Supor que $P_V[L_m((F_1 \wedge F_2)/A)] \neq \emptyset$. Pode-se afirmar então que $L_m((F_1 \wedge F_2)/A) \neq \emptyset$. Mas, uma vez que F_1 e F_2 foram obtidos a partir de especificações prefixo-fechadas, então $L_m((F_1 \wedge F_2)/A) \neq \emptyset \Rightarrow \epsilon \in L_m((F_1 \wedge F_2)/A)$. Conforme mostrado antes (nesta mesma prova), $L_m((F_1 \wedge F_2)/H) = L_m((F_1 \wedge F_2)/A) \cap L$. Assim, como $\epsilon \in L$ e $\epsilon \in L_m((F_1 \wedge F_2)/A)$, tem-se que $\epsilon \in L_m((F_1 \wedge F_2)/H)$. Desta forma, $\epsilon \in P_V[L_m((F_1 \wedge F_2)/H)]$ e portanto $P_V[L_m((F_1 \wedge F_2)/H)] \neq \emptyset$.

$\diamond$

Pode-se estabelecer agora o seguinte resultado.

Corolário 6.2 *Dado um problema de Síntese Modular de Supervisores (SMSH) para uma planta híbrida $\mathcal{H}$ e duas especificações prefixo-fechadas E_1, E_2. Sejam F_i $(i=1,2)$ supervisores projetados para solucionar este problema na abordagem SED baseado em uma aproximação A que é v-coerente para o modelo lógico exato de $\mathcal{H}$. Então, os supervisores C/E $\mathcal{F}_i$ equivalentes aos supervisores SED F_i constituem uma solução para o problema SMSH original.*

Prova: *Segue diretamente do Teorema 6.2 e do Corolário 4.1.*

$\diamond$

Este resultado significa que se os supervisores SED F_1 e F_2 consistem numa solução para um dado problema de síntese de supervisores modulares formulado a partir de uma aproximação A que seja v-coerente para H, então estes mesmos supervisores consistem também numa solução para o problema formulado para H. Além disso, os supervisores $\mathcal{F}_1$ e $\mathcal{F}_2$ obtidos através do Algoritmo 1 apresentado em (González et al., 2001) constituem uma solução correspondente para o problema original formulado para a planta híbrida $\mathcal{H}$.

6.4 Refinamento das Aproximações

Conforme comentado anteriormente, uma vez que A é apenas uma aproximação para o comportamento lógico do sistema híbrido $\mathcal{H}$, se nenhuma solução é encontrada para um dado

problema de síntese de supervisores baseado numa aproximação A, não significa que este problema não tenha solução (Niinomi et al., 1996; Cury e Krogh, 1999). Seja A' um refinamento para a aproximação A, tal que $L \subset L(A') \subset L(A)$. Tendo em vista que o refinamento de uma aproximação é feito com base na possibilidade de ocorrência de eventos e não sobre a escolha das condições associadas a estes (Niinomi et al., 1996), se a aproximação A é v-coerente para H então o modelo refinado A' também é v-coerente para H.

Assim, no contexto da abordagem monolítica, adaptando um resultado introduzido em (Cury e Krogh, 1999), pode-se concluir que se não existir um supervisor F tal que $\emptyset \neq P_V[L_m(F/A)] \subseteq E$, então refinamentos em A podem levar a uma aproximação mais refinada A' para a qual existe um supervisor F' tal que $\emptyset \neq P_V[L_m(F'/A')] \subseteq E$. Este resultado é ilustrado por intermédio do exemplo apresentado a seguir.

Exemplo 6.2 *Seja o modelo H e a aproximação externa A mostrados na Figura 6.3. Seja ainda uma especificação $E = \overline{v_1 \circ v_3}$. Então $K = P_V^{-1}(E) \cap L(A) = \overline{v_0u_1 \circ v_1u_1 \circ v_3u_1} + \overline{v_0u_1 \circ v_1u_2} + \overline{v_0u_2}$ e $Sup\mathbb{C}_{VU}(K) = \emptyset$, de onde se obtém que $Sup\mathbb{C}_V(E) = \emptyset$. Sendo assim, não existe um supervisor F para A tal que $\emptyset \neq P_V[L_m(F/A)] \subseteq E$. Seja agora A' um refinamento para a aproximação A, tal que $L \subset L(A') \subset L(A)$, conforme mostrado na Figura 6.3. Note que para A' existe um supervisor F' tal que $\emptyset \neq P_V[L_m(F'/A')] \subseteq E$. Em particular, o supervisor F' obtido neste exemplo a partir de A' é tal que $P_V[L_m(F'/A')] = E$.*

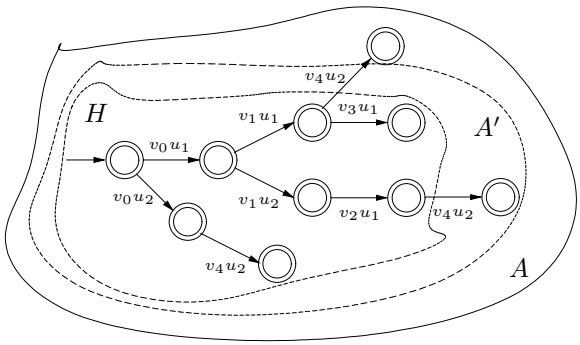


Figura 6.3: Autômato H, aproximação A e refinamento A'.

■

Seja F um supervisor obtido com base na aproximação A e F' um supervisor obtido para a aproximação refinada A'. Como A' é mais restritiva que A no sentido que $L(A') \subset L(A)$, então geralmente o supervisor F' é mais permissivo que o supervisor F, isto é, geralmente ocorre

que $P_V[L_m(F/A')] \subset P_V[L_m(F'/A')]$. Sustentado em outro resultado apresentado em (Cury e Krogh, 1999), pode-se afirmar que se um supervisor F' obtido com base em A' é tal que $P_V[L_m(F/A')] \subset P_V[L_m(F'/A')]$, então $P_V[L_m(F/H)] \subset P_V[L_m(F'/H)]$, ou seja, um supervisor menos restritivo encontrado para uma aproximação mais refinada deve ser menos restritivo também para o sistema híbrido. O exemplo apresentado a seguir ilustra esta característica.

Exemplo 6.3 *Considere agora os modelos mostrados na Figura 6.4. Seja a especificação* $E = \overline{v_1 \circ v_3} + \overline{v_1 \circ v_2 \circ v_4}$*, então a máxima linguagem vu-controlável e.r.a. L(A) é dada por* $Sup\mathbb{C}_{_{VU}}[P_V^{-1}(E) \cap L(A)] = \overline{v_0u_1 \circ v_1u_2 \circ v_2u_1 \circ v_4u_2}$*. Sendo assim, projetando um supervisor*

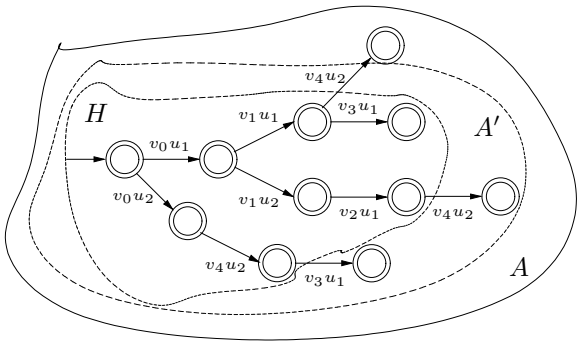
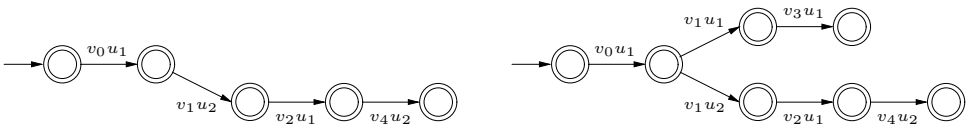


Figura 6.4: Autômato H, aproximação A e refinamento A′.

F com base em A obtém-se $L_m(F/A) = \overline{v_0u_1 \circ v_1u_2 \circ v_2u_1 \circ v_4u_2}$ *e portanto* $P_V[L_m(F/A)] = \overline{v_1 \circ v_2 \circ v_4}$*. Note ainda que* $P_V[L_m(F/A')] = P_V[L_m(F/A) \cap L(A')] = \overline{v_1 \circ v_2 \circ v_4}$*. Fazendo agora o projeto do supervisor F′ a partir de A′, obtém-se* $L_m(F'/A') = Sup\mathbb{C}_{_{VU}}[P_V^{-1}(E) \cap L(A')] = \overline{v_0u_1 \circ v_1u_1 \circ v_3u_1} + \overline{v_0u_1 \circ v_1u_2 \circ v_2u_1 \circ v_4u_2}$*, e assim* $P_V[L_m(F'/A')] = \overline{v_1 \circ v_3} + \overline{v_1 \circ v_2 \circ v_4}$*. Note que* $P_V[L_m(F/A')] \subset P_V[L_m(F'/A')]$*. Na Figura 6.5 mostra-se o supervisor F obtido a partir de A e também o supervisor F′ obtido a partir de A′.*



(a) Supervisor F (b) Supervisor F'

Figura 6.5: Supervisores encontrados para o Exemplo 6.3.

Observe ainda que $P_V[L_m(F/H)] = P_V[L_m(F/A) \cap L] = \overline{v_1 \circ v_2}$ *e que* $P_V[L_m(F'/H)] = P_V[L_m(F'/A') \cap L] = \overline{v_1 \circ v_3 + v_1 \circ v_2}$ *e que portanto* $P_V[L_m(F/H)] \subset P_V[L_m(F'/H)]$*.* ■

No contexto da abordagem modular, suponha que a síntese modular de supervisores seja realizada com base numa aproximação A, seguindo o procedimento adotado no Capítulo 5, e que não existam supervisores F_1 e F_2 tais que $\emptyset \neq P_V[L_m((F_1 \wedge F_2)/A)] \subseteq E_1 \cap E_2$. O uso de um autômato aproximação A', obtido através do refinamento de A, para o qual $L \subseteq L(A') \subseteq L(A)$, pode levar a supervisores F_1' e F_2' tais que $\emptyset \neq P_V[L_m((F_1' \wedge F_2')/A')] \subseteq E_1 \cap E_2$, conforme mostrado no exemplo que segue.

Exemplo 6.4 *Considere agora os modelos mostrados na Figura 6.6. Sejam duas especificações* $E_1, E_2 \subseteq P_V[L(A)]$ *dadas por* $E_1 = \overline{v_2 \circ v_3 \circ v_4 \circ v_5 + v_4 \circ v_6 \circ v_2 + v_4 \circ v_6 \circ v_3}$ *e* $E_2 = \overline{v_1 + v_2 \circ v_3 \circ v_4 \circ v_1 + v_4 \circ v_6 \circ v_2}$. *Utilizando o procedimento detalhado no Capítulo*

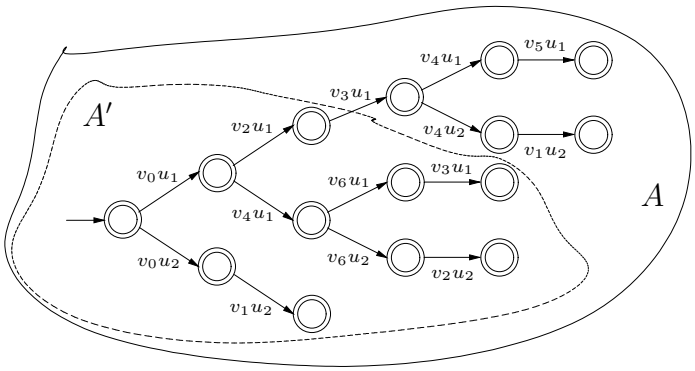
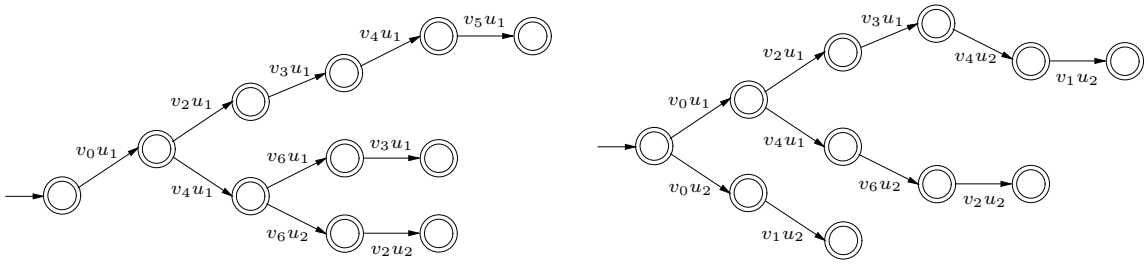


Figura 6.6: Autômato H, aproximação A e refinamento A'.

5, a partir destas especificações determina-se $K_1{}^{\uparrow} = Sup\mathbb{C}_{VU}[P_V^{-1}(E_1) \cap L(A)]$ *e* $K_2{}^{\uparrow} = Sup\mathbb{C}_{VU}[P_V^{-1}(E_2) \cap L(A)]$. *Autômatos que reconhecem estas linguagens são mostrados na Figura 6.7.*

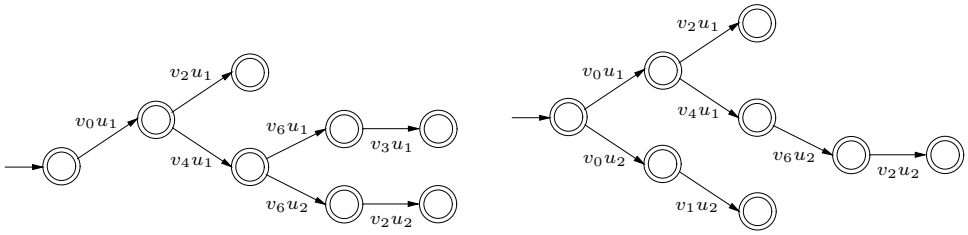


(a) $Sup\mathbb{C}_{VU}(P_V^{-1}(E_1) \cap L(A))$ (b) $Sup\mathbb{C}_{VU}(P_V^{-1}(E_2) \cap L(A))$

Figura 6.7: Projeto dos supervisores com base em A.

Pode-se verificar que $Sup\mathcal{IC}_{VU}(K_1{}^{\uparrow}, K_2{}^{\uparrow}) = \emptyset$ *e que portanto, de acordo com o Teorema 5.1, não existem dois supervisores,* F_1, F_2, *tais que* $\emptyset \neq P_V[L_m((F_1 \wedge F_2)/A)] \subseteq E_1 \cap E_2$.

Seja agora uma aproximação A' mais refinada do que A, conforme mostrado na Figura 6.6. Projetando novos supervisores F_1' e F_2' com base nesta aproximação A' obtém-se $L_m(F_1'/A') = Sup\mathbb{C}_{VU}[P_V^{-1}(E_1) \cap L(A')]$ e $L_m(F_2'/A') = Sup\mathbb{C}_{VU}[P_V^{-1}(E_2) \cap L(A')]$ (ver Figura 6.8).



(a) $Sup\mathbb{C}_{VU}(P_V^{-1}(E_1) \cap L(A'))$ (b) $Sup\mathbb{C}_{VU}(P_V^{-1}(E_2) \cap L(A'))$

Figura 6.8: Projeto dos supervisores com base em A'.

Pode-se facilmente verificar que estas linguagens são interconsistentes, e assim tem-se que $F_1' \wedge F_2'$ é um supervisor consistente para A' tal que $L_m((F_1' \wedge F_2')/A') = Sup\mathbb{C}_{VU}[P_V^{-1}(E_1) \cap L(A')] \cap Sup\mathbb{C}_{VU}[P_V^{-1}(E_2) \cap L(A')] = \overline{v_0u_1 \circ v_2u_1 + v_0u_1 \circ v_4u_1 \circ v_6u_2 \circ v_2u_2}$. Logo, $P_V[L_m((F_1' \wedge F_2')/A')] = \overline{v_2 + v_4 \circ v_6 \circ v_2} \neq \emptyset$.

■

6.5 Conclusões

Neste capítulo estudaram-se os problemas que surgem quando o projeto dos supervisores (seja na abordagem monolítica seja na abordagem modular) é feito com base em uma aproximação para o comportamento do sistema híbrido. Mostrou-se que o problema de bloqueio é difícil de ser solucionado e então tratou-se somente de problemas cujas especificações são prefixo-fechadas.

No contexto da abordagem monolítica, mostrou-se que um supervisor solução para um dado PCSE formulado a partir de uma aproximação A que é v-coerente para H, é também uma solução para o PCSE formulado para a planta H. Além disso, o supervisor $\mathcal{F}$ obtido através do Algoritmo 1 apresentado em (González et al., 2001) é uma solução correspondente para o problema original formulado para a planta híbrida $\mathcal{H}$.

De forma semelhante, no contexto da abordagem modular verificou-se que se os supervisores SED F_1 e F_2 consistem numa solução para um dado problema de síntese de supervisores modulares formulado a partir de uma aproximação A que é v-coerente para H, então estes

mesmos supervisores consistem também numa solução para o problema formulado para H. Além disso, os supervisores $\mathcal{F}_1$ e $\mathcal{F}_2$ obtidos através do Algoritmo 1 apresentado em (González et al., 2001) constituem uma solução correspondente para o problema original formulado para a planta híbrida $\mathcal{H}$.

Por outro lado, uma vez que A é apenas uma aproximação para o comportamento lógico do sistema híbrido $\mathcal{H}$, se nenhuma solução é encontrada para um dado problema de síntese de supervisores (na abordagem monolítica ou na modular) baseado numa aproximação A, não significa que este problema não tenha solução. Constatou-se que se o projeto dos supervisores for feito com base em uma aproximação A' que é mais próxima de H (mais refinada do que A), então é possível que se encontre uma solução para este problema. Por este motivo, é importante que se possa obter aproximações que sejam tão próximas de H quanto possível.

Capítulo 7

Conclusões

A apresentação deste capítulo está dividida em duas seções. Na Seção 7.1 descrevem-se rapidamente os pontos abordados e as principais contribuições desta tese. Na Seção 7.2 apresentam-se algumas perspectivas de extensão dos resultados obtidos neste trabalho.

7.1 Tópicos Abordados e Principais Contribuições

Neste trabalho, tratou-se das abordagens monolítica e modular para o controle supervisório de uma classe de sistemas híbridos. Os sistemas híbridos considerados possuem dinâmicas contínuas e discretas interagindo entre si, sendo que a dinâmica discreta é definida por eventos gerados quando variáveis do espaço de estados contínuo (ou funções destas) alcançam certos patamares, forçando então transições no estado discreto. A dinâmica contínua, por sua vez, é determinada por uma condição discreta, função do estado discreto atual do sistema.

Apresentou-se um problema de controle supervisório para sistemas híbridos (SSSH) utilizando-se o paradigma de modelagem de sistemas condição/evento, e mostrou-se que este problema pode ser solucionado por intermédio da resolução de um problema equivalente no domínio discreto, denominado Problema de Controle Supervisório Equivalente (PCSE). O problema aqui considerado consiste em uma generalização daquele abordado nos trabalhos de Cury et al. (1998), Koutsoukos et al. (2000) e Moor et al. (1998), nos quais o caráter discreto advém somente do supervisor.

Propôs-se uma abordagem modular para a síntese de supervisores para sistemas híbridos e mostrou-se que, diferentemente da teoria clássica de controle modular de SEDs, para os

sistemas híbridos a modularidade entre as especificações (linguagens) não é suficiente para garantir que a ação conjunta dos supervisores modulares leve o sistema sob supervisão a ter um comportamento não bloqueante e minimamente restritivo, conforme desejado. Introduziu-se então o conceito de *interconsistência* entre linguagens e estabeleceram-se condições sob as quais se pode utilizar a abordagem modular para o controle supervisório de sistemas híbridos.

Propôs-se ainda uma metodologia para a síntese de supervisores modulares que sempre leva a uma solução ótima. Desta forma, se existe uma solução para um determinado problema de controle supervisório para sistemas híbridos por intermédio da abordagem monolítica, então o mesmo resultado certamente será obtido por intermédio da abordagem modular.

Exemplos foram utilizados para ilustrar a metodologia proposta, sendo que nestes exemplos todo processo de síntese dos supervisores modulares foi feito com auxílio da ferramenta GRAIL, tendo-se utilizado funções desenvolvidas especificamente para este propósito. Utilizou-se ainda a ferramenta CHECKMATE e o ambiente SIMULINK/MATLAB. Mostrou-se então todo o procedimento desde a modelagem, passando pela obtenção de um autômato aproximação para o comportamento lógico do sistema híbrido, obtido por intermédio do CHECKMATE, resolvendo o problema no domínio de SEDs com auxílio do GRAIL e, por fim, com auxílio do SIMULINK/MATLAB, fazendo uma simulação do comportamento do sistema em malha fechada.

Estudou-se ainda os problemas que surgem quando o projeto dos supervisores (seja na abordagem monolítica ou na abordagem modular) é feito com base em uma aproximação para o comportamento do sistema híbrido. Mostrou-se que o problema de bloqueio é difícil de ser solucionado e então tratou-se somente de problemas cujas especificações são prefixo-fechadas. Introduziu-se o conceito de *aproximação v-coerente* e mostrou-se que se um supervisor é uma solução para um problema formulado para uma aproximação v-coerente para o modelo da planta híbrida, então este supervisor é também uma solução para o problema original (para a planta híbrida), sendo este resultado válido tanto para a abordagem monolítica quanto para a modular. Por outro lado, uma vez que A é apenas uma aproximação para o comportamento lógico do sistema híbrido $\mathcal{H}$, se nenhuma solução é encontrada para um dado problema de síntese de supervisores (na abordagem monolítica ou na modular) baseado numa aproximação A, não significa que este problema não tenha solução. Mostrou-se que se o projeto dos supervisores for feito com base em uma aproximação A' que é mais próxima de H (mais refinada do que A), então é possível que se encontre uma solução para este problema. Por este motivo, é importante que se possa obter aproximações que sejam tão próximas de H quanto possível.

De forma bastante sucinta, pode-se dizer que as principais contribuições desta tese foram:

- a consolidação da abordagem monolítica para a síntese de supervisores para sistemas híbridos e a extensão da mesma para linguagens marcadas;
- a proposição de uma metodologia para a síntese modular de supervisores que garante a otimalidade da solução; e
- o estabelecimento de uma condição sob a qual se pode garantir que supervisores sintetizados com base em um modelo aproximado para a planta híbrida sejam uma solução também para o problema original.

7.2 Perspectivas para Futuros Trabalhos

De uma forma geral, tanto no contexto de sistemas a eventos discretos quanto no contexto de sistemas híbridos, a abordagem modular possui algumas vantagens em relação à abordagem monolítica. Uma destas vantagens consiste na diminuição da complexidade computacional da síntese dos supervisores, uma vez que os mesmos são obtidos para especificações menores que a usada na abordagem monolítica.

Entretanto, da mesma forma que na abordagem monolítica, na síntese dos supervisores modulares utiliza-se um modelo que representa o comportamento global para a planta. Assim, quando este modelo possui um elevado número de estados, a complexidade computacional continua sendo um problema importante, e portanto um ponto negativo de ambas as abordagens.

No intuito de aperfeiçoar a abordagem modular de síntese de supervisores para SEDs, de Queiroz (2004) propôs uma abordagem que permite a obtenção de um supervisor ótimo para cada especificação fazendo apenas a composição dos subsistemas diretamente afetados por ela, sem a necessidade portanto de obter um modelo global para a planta. Chamada de abordagem de controle modular local, esta metodologia pode reduzir consideravelmente a complexidade computacional da síntese dos supervisores modulares mesmo quando a planta apresenta um modelo complexo.

Pretende-se estender os resultados obtidos nesta tese através da adaptação da abordagem de controle modular local para a síntese de supervisores para sistemas híbridos.

Deve-se fazer ainda um estudo comparativo da complexidade computacional para estas abordagens, de forma a mostrar mais claramente as vantagens e desvantagens de cada uma delas.

Referências Bibliográficas

Altafini, C., Speranzon, A. e Johansson, K. H. (2002). Hybrid control of a truck and trailer vehicle, *in* Tomlin e Greenstreet (2002), pp. 21–34. Proceedings of the 5th International Workshop, HSCC 2002, Stanford, CA, USA, March 25-27, 2002.

Alur, R., Courcoubetis, C. e Dill, D. (1990). Model-checking for real-time systems, *Proceedings of the 5th IEEE Symposium on Logic in Computer Science*, pp. 414–425.

Alur, R., Courcoubetis, C., Halbwachs, N., Henzinger, T. A., Ho, P.-H., Nicollin, X., Olivero, A., Sifakis, J. e Yovine, S. (1995). The algorithmic analysis of hybrid systems, *Theoretical Computer Science* **138**(1): 3–34.

Alur, R., Courcoubetis, C., Henzinger, T. A. e Ho, P.-H. (1993). Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems, *in* Grossman et al. (1993), pp. 209–229.

Alur, R., Dang, T., Esposito, J. M., Hur, Y., Ivancic, F., Kumar, V., Lee, I., Mishra, P., Pappas, G. J. e Sokolsky, O. (2003). Hierarchical modeling and analysis of embedded systems., *Proceedings of the IEEE* **91**(1): 11–28.

Alur, R. e Dill, D. L. (1990). Automata for modeling real-time systems, *in* M. S. Paterson (ed.), *Proc. of the 17th International Colloquium on Automata, Languages and Programming, ICALP90*, Vol. 443 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 322–335.

Alur, R. e Dill, D. L. (1994). A theory of timed automata, *Theoretical Computer Science* **126**(2): 183–235.

Alur, R., Grosu, R., Hur, Y., Kumar, V. e Lee, I. (2000). Modular specification of hybrid systems in CHARON, *in* Lynch e Krogh (2000), pp. 6–19. Proceedings of the Third International Workshop, HSCC 2000, Pittsburgh, PA, USA, March 23-25, 2000.

Alur, R., Grosu, R., Lee, I. e Sokolsky, O. (2001). Compositional refinement for hierarchical hybrid systems, *in* Benedetto e Sangiovanni-Vincentelli (2001), pp. 33–48. Proceedings of the 4th International Workshop, HSCC 2001, Rome, Italy, March 28-30, 2001.

Alur, R., Henzinger, T. A. e Ho, P.-H. (1996). Automatic symbolic verification of embedded systems, *IEEE Transactions on Software Engineering* **22**(3): 181–201.

Alur, R., Henzinger, T. A. e Sontag, E. D. (eds) (1996). *Hybrid Systems III: Verification and Control*, Vol. 1066 of *Lecture Notes in Computer Science*, Springer-Verlag.

Alur, R. e Pappas, G. J. (eds) (2004). *Hybrid Systems: Computation and Control, 7th International Workshop, HSCC 2004, Philadelphia, PA, USA, March 25-27, 2004, Proceedings*, Vol. 2993 of *Lecture Notes in Computer Science*, Springer.

Amla, N., Kurshan, R., McMillan, K. e Medel, R. (2003). Experimental analysis of different techniques for bounded model checking, *in* H. Garavel e J. Hatcliff (eds), *Ninth International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, Vol. 2619 of *LNCS*, Springer-Verlag GmbH, Poland, pp. 34–48.

Andersson, M. (1994). *Object-Oriented Modeling and Simulation of Hybrid Systems*, Ph.D. Thesis, Lund Institute of Technology, Sweden.

Andronov, A. e Chaikin, C. (1949). *Theory of Oscillations*, Princeton Univ. Press, Princeton, NJ.

Antsaklis, P. J. (2000a). A brief introduction to the theory and applications of hybrid systems, *Proceedings of the IEEE, Special Issue on Hybrid Systems: Theory and Applications* **88**(7): 879–887.

Antsaklis, P. J. (ed.) (2000b). *Special Issue on Hybrid Systems: Theory and Applications*, Vol. 88, No. 7 of *Proceedings of the IEEE*.

Antsaklis, P. J., Kohn, W., Lemmon, M., Nerode, A. e Sastry, S. (eds) (1999). *Hybrid Systems V*, Vol. 1567 of *Lecture Notes in Computer Science*, Springer-Verlag.

Antsaklis, P. J. e Koutsoukos, X. D. (2002). Hybrid systems control, *Encyclopedia of Physical Science and Technology, Third Edition* **7**: 445–458. Also as ISIS Technical Report ISIS-2001-003, February 2001.

Antsaklis, P. J. e Koutsoukos, X. D. (2003). Hybrid systems: Review and recent progress, *in* T. Samad e G. Balas (eds), *Software-Enabled Control: Information Technology for Dynamical Systems*, IEEE Press, pp. 273–298.

Antsaklis, P. J., Koutsoukos, X. D. e Zaytoon, J. (1998). On hybrid control of complex systems: A survey, *European Journal of Automation, APII-JESA, Journal europeen des systemes automatises* **32**(9-10): 1023–1045.

Antsaklis, P. J. e Lemmon, M. D. (1998a). Introduction to the special issue on hybrid systems, *Discrete Event Dynamic Systems: Theory and Applications, Special Issue on Hybrid Systems* **8**(2): 101–103.

Antsaklis, P. J. e Lemmon, M. D. (eds) (1998b). *Special Issue on Hybrid Systems*, Vol. 8, No. 2 of *Journal of Discrete Event Dynamical Systems: Theory and Applications.*

Antsaklis, P. J. e Nerode, A. (1998a). Hybrid control systems: An introductory discussion to the special issue, *IEEE Transactions on Automatic Control, Special Issue on Hybrid Control Systems* **43**(4): 457–460.

Antsaklis, P. J. e Nerode, A. (eds) (1998b). *Special Issue on Hybrid Control Systems, IEEE Transactions on Automatic Control*, Vol. 43, No. 4.

Antsaklis, P. J., Stiver, J. A. e Lemmon, M. D. (1993). Hybrid system modeling and autonomous control systems, *in* Grossman et al. (1993), pp. 366–392.

Antsaklis, P., Kohn, W., Nerode, A. e Sastry, S. (eds) (1995). *Hybrid Systems II*, Vol. 999 of *Lecture Notes in Computer Science*, Springer-Verlag.

Antsaklis, P., Kohn, W., Nerode, A. e Sastry, S. (eds) (1997). *Hybrid Systems IV*, Vol. 1273 of *Lecture Notes in Computer Science*, Springer-Verlag.

Asarin, E., Dang, T. e Maler, O. (2002). The d/dt Tool for Verification of Hybrid Systems, *Proceedings of the 14th International Conference on Computer Aided Verification – CAV'2002, Copenhagen, Denmark*, Vol. 2404 of *LNCS*, Springer-Verlag, pp. 365–370.

Ästrom, K. J. e Wittenmark, B. (1997). *Computer Controled Systems: Theory and Design*, 3rd edn, Prentice Hall.

Avraam, M. P., Shah, N. e Pantelides, C. C. (1998). Modelling and optimization of general hybrid systems in the continuous time domain, *Computers and Chemical Engineering* **22**. Suppl., S221-S228.

Balluchi, A., Benvenuti, L., Di Benedetto, M. D., Pinello, C. e Sangiovanni-Vincentelli, A. L. (2000). Automotive engine control and hybrid systems: Challenges and opportunities, *Proceedings of the IEEE, Special Issue on Hybrid Systems: Theory and Applications* **88**(7): 888–912.

Barton, P. I. (1992). *The Modelling and Simulation of Combined Discrete/Continuous Processes*, Ph.D. Thesis, University of London.

Behrmann, G., David, A. e Larsen, K. G. (2004). A tutorial on UPPAAL, *in* M. Bernardo e F. Corradini (eds), *Formal Methods for the Design of Real-Time Systems: 4th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM-RT 2004*, number 3185 in *LNCS*, Springer–Verlag, pp. 200–236.

Bemporad, A. e Morari, M. (1999). Control of systems integrating logic, dynamics, and constraints, *Automatica* **35**(3): 407–427. Special issue on hybrid systems.

Benedetto, M. D. D. e Sangiovanni-Vincentelli, A. L. (eds) (2001). *Hybrid Systems: Computation and Control*, Vol. 2034 of *Lecture Notes in Computer Science*, Springer-Verlag. Proceedings of the 4th International Workshop, HSCC 2001, Rome, Italy, March 28-30, 2001.

Benveniste, A. e Berry, G. (1991). The synchronous approach to reactive and real-time systems, *Proceedings of the IEEE, Special Issue on Reactive and Real-Time Systems* **79**(9): 1270–1282.

Benveniste, A. e Guernic, P. L. (1990). Hybrid dynamical systems theory and the SIGNAL language, *IEEE Transactions on Automatic Control* **35**(5): 535–546.

Bicchi, A. e Pallottino, L. (2000). On optimal cooperative conflict resolution for air traffic management systems, *IEEE Transactions on Intelligent Transportation Systems* **1**(4): 221–231.

Bjorner, N., Browne, A., Chang, E., Colon, M., Kapur, A., Manna, Z., Sipma, H. e Uribe, T. (1995). STeP: The Stanford Temporal Prover, user's manual, *Technical Report STAN-CS-TR-95-1562*, Dept. of Computer Science, Stanford University.

Ábrahám, E., Becker, B., Klaedtke, F. e Steffen, M. (2005). Optimizing bounded model checking for linear hybrid systems, *Proceedings of the 6th International Workshop on Verification, Model Checking, and Abstraction (VMCAI'05)*, number 3885 in *LNCS*, Springer-Verlag, Paris, pp. 396–412.

Brandin, B. A. (1996). The real-time supervisory control of an experimental manufacturing cell, *IEEE Transactions on Robotics and Automation* **12**(1): 1–14.

Branicky, M. S. (1995). *Studies in Hybrid Systems: Modeling, Analysis, and Control*, Ph.D. Thesis, Massachusetts Institute of Technology - MIT, Department of Electrical Engineering and Computer Science, Cambridge.

Branicky, M. S. (1998). Multiple Lyapunov functions and other analysis tools for switched and hybrid systems, *IEEE Transactions on Automatic Control, Special Issue on Hybrid Control Systems* **43**(4): 475–482.

Branicky, M. S., Borkar, V. S. e Mitter, S. K. (1994). A unified framework for hybrid control, *Proceedings of the 33rd IEEE Conference on Decision and Control*, Lake Buena Vista, FL, pp. 4228–4234.

Branicky, M. S., Borkar, V. S. e Mitter, S. K. (1998). A unified framework for hybrid control: Model and optimal control theory, *IEEE Transactions on Automatic Control* **43**(1): 31–45.

Branicky, M. S. e Chhatpar, S. R. (2002). A computational framework for the verification and synthesis of force-guided robotic assembly strategies, *in* Tomlin e Greenstreet (2002), pp. 120–133. Proceedings of the 5th International Workshop, HSCC 2002, Stanford, CA, USA, March 25-27, 2002.

Brooks, C., Cataldo, A., Lee, E. A., Liu, J., Liu, X., Neuendorffer, S. e Zheng, H. (2004). HyVisual: A Hybrid System Visual Modeler, *Technical Memorandum UCB/ERL M04/18*, University of California, Berkeley.

Brunetti, F. (2004). *Mecanica dos Fluidos*, 1 edn, Pearson Education (Prentice–Hall).

Caines, P. E. e Wei, Y.-J. (1998). Hierarchical hybrid control systems: A lattice theoretic formulation, *IEEE Transactions on Automatic Control, Special Issue on Hybrid Control Systems* **43**(4): 501–508.

Carloni, L., Benedetto, M. D. D., Passerone, R., Pinto, A. e Sangiovanni-Vincentelli, A. (2004). Modeling techniques, programming languages and design toolsets for hybrid systems, *Project IST-2001-38314 – Design of Embedded Controllers for Safety Critical Systems (Columbus) DHS4-5-6*, Information Society Technologies – IST.

Cellier, F. E. (1979). *Combined Continuous/Discrete System Simulation by Use of Digital Computers: Techniques and Tools*, Ph.D. Thesis, Swiss Federal Institute of Technology, Zurich, Switzerland.

Chase, C., Serrano, J. e Ramadge, P. J. (1993). Periodicity and chaos from switched flow systems: Contrasting examples of discretely controlled continuous systems, *IEEE Transactions on Automatic Control* **38**(1): 70–83.

Chutinan, A. (1999). *Hybrid System Verification Using Discrete Model Approximations*, Ph.D. Thesis, Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, USA.

Chutinan, A. e Krogh, B. H. (1999a). Computing approximating automata for a class of linear hybrid systems, *in* Antsaklis et al. (1999), pp. 16–37.

Chutinan, A. e Krogh, B. H. (1999b). Verification of polyhedral-invariant hybrid automata using polygonal flow pipe approximations, *in* Vaandrager e van Schuppen (1999), pp. 76–90. Proceedings of the Second International Workshop, HSCC'99, Berg en Dal, The Netherlands, March 29-31, 1999.

Chutinan, A. e Krogh, B. H. (2001). Infinite-state transition system verification using approximate quotient transition systems, *IEEE Transactions on Automatic Control* **46**(9): 1401–1410.

Chutinan, A. e Krogh, B. H. (2003). Computational techniques for hybrid system verification, *IEEE Transactions on Automatic Control* **48**(1): 64–75.

Clarke, E. M., Fehnker, A., Han, Z., Krogh, B., Stursberg, O. e Theobald, M. (2003). Verification of hybrid systems based on counterexample-guided abstraction refinement, *in* H. Garavel e J. Hatcliff (eds), *Ninth International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, Vol. 2619 of *LNCS*, Springer-Verlag GmbH, Poland, pp. 192–207.

Corona, D. (2005). *Optimal Control of Linear Affine Hybrid Automata*, Ph.D. Thesis, University of Cagliari, Italy.

Cury, J. E. R. e Garcia, T. R. (2004). Uma abordagem de sistemas híbridos para modelagem e verificação de sistemas de tráfego urbano, *Proceedings of the XIII Pan-American Conference of Traffic and Transportation Engineering*, Vol. 1, pp. 1–12.

Cury, J. E. R. e Krogh, B. H. (1999). Synthesizing supervisory controllers for hybrid systems, *Journal of the Society of Instrument and Control Engineers (SICE)* **38**(3): 161–168.

Cury, J. E. R., Krogh, B. H. e Niinomi, T. (1998). Synthesis of supervisory controllers for hybrid systems based on approximating automata, *IEEE Transactions on Automatic Control, Special Issue on Hybrid Control Systems* **43**(4): 564–568.

Cury, J. E. R., Torrico, C. R. C. e da Cunha, A. E. C. (2001). A new approach for supervisory control of discrete event systems, *Proceedings of the European Control Conference – ECC'01*, Porto, Portugal, pp. 1595–1599.

Cury, J. E. R., Torrico, C. R. C. e da Cunha, A. E. C. (2004). Supervisory control of discrete event systems with flexible marking, *European Journal of Control* **10**(1): 47–60.

da Cunha, A. E. C. (2003). *Contribuições ao Controle Hierárquico de Sistemas a Eventos Discretos*, Tese (doutorado), Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Santa Catarina, Florianópolis, SC.

da Cunha, A. E. C. e Cury, J. E. R. (2002). Hierarchically consistent controlled discrete event systems, *Proceedings of the 15th IFAC World Congress on Automatic Control*, Vol. 15, Barcelona, Espanha, pp. 1–6.

da Silva Jr., B. I. (2004). *Modeling and Verification of Hybrid Systems with Clocked and Unclocked Events*, Ph.D. Thesis, Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, USA.

Dang, T. (2000). *Verification and Synthesis of Hybrid Systems*, Ph.D. Thesis, Institut National Polytechnique de Grenoble, Verimag, France.

David, R. (1997). Modeling of hybrid systems using continuous and hybrid petri nets, *Proceedings of the 6th International Workshop on Petri Nets and Performance Models PNPM97*, St-Malo, France, pp. 47–57.

David, R. e Alla, H. (2001). On hybrid petri nets, *Discrete Event Dynamic Systems: Theory and Applications* **11**: 9–40.

Davis, J., Goel, M., Hylands, C., Kienhuis, B., Lee, E. A., Liu, J., Liu, X., Muliadi, L., Neuendorker, S., Reekie, J., Smyth, N., Tsay, J. e Xiong, Y. (1999). Overview of the Ptolemy project, *Technical Report UCB/ERL M99/37*, University of California, Berkeley.

Davoren, J. M. (1998). *Modal Logics for Continuous Dynamics*, Ph.D. Thesis, Department of Mathematics, Cornell University.

Davrazos, G. N. e Koussoulas, N. T. (2001). A review of stability results for switched and hybrid systems, *9th Mediterranean Conference on Control and Automation*, Croatia, pp. 1–8.

de Bakker, J. W., Huizing, C., de Roever, W. P. e Rozenberg, G. (eds) (1992). *Real-Time: Theory in Practice*, Vol. 600 of *Lecture Notes in Computer Science*, Springer. Proceedings of the REX Workshop, June 3-7, 1991, Mook, The Netherlands.

de Paula e Silva, D. (2004). *Modelagem, análise e controle supervisório de sistemas híbridos em uma planta piloto*, Dissertação (mestrado), Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Santa Catarina, Florianópolis, SC.

de Queiroz, M. H. (2004). *Controle Supervisório Modular e Multitarefa de Sistemas Compostos*, Tese (doutorado), Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Santa Catarina, Florianópolis, SC.

DeCarlo, R. A., Branicky, M. S., Pettersson, S. e Lennartson, B. (2000). Perspectives and results on the stability and stabilizability of hybrid systems, *Proceedings of the IEEE, Special Issue on Hybrid Systems: Theory and Applications* **88**(7): 1069–1082.

Demongodin, I. e Koussoulas, N. T. (1998). Differential petri nets: representing continuous systems in a discrete-event world, *IEEE Transactions on Automatic Control, Special Issue on Hybrid Control Systems* **43**(4): 573–579.

Deshpande, A. (1994). *Control of Hybrid Systems*, Ph.D. Thesis, University of California at Berkeley, Berkeley, California.

Deshpande, A., Göllü, A. e Semenzato, L. (1998). The SHIFT programming language for dynamic networks of hybrid automata, *IEEE Transactions on Automatic Control, Special Issue on Hybrid Control Systems* **43**(4): 584–587.

Deshpande, A., Göllü, A. e Varaiya, P. (1997). SHIFT: A Formalism and a Programming Language for Dynamic Networks of Hybrid Automata, *in* Antsaklis et al. (1997), pp. 113–133.

Djenini, R. (2001). *Formalisme de modélisation des systèmes hybrides*, Ph.D. Thesis, Institut National de Recherche en Informatique et en Automatique - INRIA, Universite Paris XII – Val de Marne.

Egerstedt, M. (2000). Behavior based robotics using hybrid automata, *in* Lynch e Krogh (2000), pp. 103–116. Proceedings of the Third International Workshop, HSCC 2000, Pittsburgh, PA, USA, March 23-25, 2000.

Egerstedt, M. e Hu, X. (2002). A hybrid control approach to action coordination for mobile robots, *Automatica* **38**(1): 125–130.

Eker, J., Janneck, J. W., Lee, E. A., Liu, J., Ludwig, J., Neuendorffer, S., Sachs, S. e Xiong, Y. (2003). Taming heterogeneity - The Ptolemy approach, *Proceedings of the IEEE* **91**(1): 127–144.

Engell, S. (ed.) (2003). *Control Engineering Practice Journal, Special Issue on Hybrid Dynamic Systems*, Vol. 12, No. 10, International Federation on Automatic Control – IFAC Conference on Analysis and Design of Hybrid Systems (ADHS'03), Saint-Malo, France.

Engell, S., Kowalewski, S., Schulz, C. e Stursberg, O. (2000). Continuous-discrete interactions in chemical processing plants, *Proceedings of the IEEE, Special Issue on Hybrid Systems: Theory and Applications* **88**(7): 1050–1068.

Ernberg, P., Fredlund, L.-A., Hansson, H., Jonsson, B., Orava, F. e Pehrson, B. (1991). Guidelines for specification and verification of communication protocols, *Technical report*, Swedwish Institute of Computer Science.

Evans, R. J. e Savkin, A. V. (eds) (1999). *Special Issue on Hybrid Control Systems*, Vol. 38, Issue 3 of *Systems and Control Letters*, Elsevier Science B.V. October 1999.

Ezzine, J. e Haddad, A. H. (1989). Controllability and observability of hybrid systems, *Proceedings of the International Journal of Control* **49**(6): 2045–2055.

Fahrland, D. A. (1970). Combined discrete event continuous systems simulation, *Simulation* **14**(2): 61–72.

Fang, L., Lin, H. e J.Antsaklis, P. (2004). Stabilization and performance analysis for a class of switched systems, *Proceedings of the 43rd Conference on Decision and Control*, IEEE, Paradise Island, The Bahamas, pp. 3265–3270.

Fábián, G. (1999). *A Language and Simulator for Hybrid Systems*, Ph.D. Thesis, Eindhoven University of Technology.

Fehnker, A. (2002). *Citius, Vilius, Melius - Guiding and Cost-Optimality in Model Checking of Timed and Hybrid Systems*, Ph.D. Thesis, Katholieke Universiteit Nijmegen, KUNijmegen.

Ferrari-Trecate, G., Gallestey, E., Letizia, P., Spedicato, M., Morari, M. e Antoine, M. (2002). Modeling and control of co-generation power plants: A hybrid system approach, *in* Tomlin e Greenstreet (2002), pp. 209–224. Proceedings of the 5th International Workshop, HSCC 2002, Stanford, CA, USA, March 25-27, 2002.

Filippov, A. F. (1988). *Differential Equations with Discontinuous Right Hand Sides*, Mathematics and Its Applications, Kluwer Academic Publishers.

Fox, R. W. e McDonald, A. T. (2001). *Introdução à Mecanica dos Fluidos*, 5 edn, LTC – Livros Técnicos e Científicos.

Franklin, G. F., Powell, J. D. e Workman, M. L. (1990). *Digital Control of Dynamic Systems*, 2nd edn, Addison-Wesley.

Frehse, G. (2005a). *Compositional Verification of Hybrid Systems using Simulation Relations*, Ph.D. Thesis, Radbout Universiteit Nijmegen.

Frehse, G. (2005b). PHAVer: Algorithmic Verification of Hybrid Systems Past HyTech, *in* Morari e Thiele (2005), pp. 258–273.

Fritz, M., Liefeldt, A. e Engell, S. (1999). Recipe-driven batch processes: Event handling in hybrid system simulation, *In Proc. of 1999 IEEE int. Symposium on Computer Aided Control System Design (CACSD'99)* pp. 138–143.

Fritzson, P. (2004). *Principles of Object-Oriented Modeling and Simulation with Modelica 2.1*, Wiley-IEEE Press.

Garcia, T. R. e Cury, J. E. R. (2002). Controle Supervisório de Sistemas Condição/Evento, *Anais do XIV Congresso Brasileiro de Automática – CBA2002*, Natal, RN, pp. 449–454.

Garcia, T. R. e Cury, J. E. R. (2004). Modelagem e verificação de sistemas de tráfego urbano através de autômatos híbridos, *Anais do Congresso Brasileiro de Automática – CBA2004*, Gramado, RS, pp. 704–709.

Girard, A. (2004). *Analyse algorithmique des systèmes hybrides*, Ph.D. Thesis, Institut National Polytechnique de Grenoble, Grenoble, France.

Girard, A. R., de Souza, J. B., Misener, J. A. e Hedrick, J. K. (2001). A control architecture for integrated cooperative cruise control and collision warning systems, *Proceedings of the Control and Decision Conference*, Florida, pp. 1–6.

Godbole, D. N. e Lygeros, J. (1994). Longitudinal control of the lead car of a platoon, *IEEE Transactions on Vehicular Technology* **43**(4): 1125–1135.

Göllü, A. e Kourjanski, M. (1997). Object-oriented design of automated highway simulations using SHIFT programming language, *Proceedings of the IEEE Conference on Intelligent Transportation Systems - ITSC '97*, Boston, MA, pp. 1–6.

Göllü, A. e Varaiya, P. (1989). Hybrid dynamical systems, *Proceedings of the 28th IEEE Conference on Decision and Control*, Vol. 3, Tampa, Florida, pp. 2708–2712.

González, J. M. E. (2000). *Aspectos de Síntese de Supervisores para Sistemas a Eventos Discretos e Sistemas Híbridos*, Tese (doutorado), Programa de Pós Graduação em Engenharia Elétrica, Universidade Federal de Santa Catarina, Florianópolis, Brasil.

González, J. M. E., da Cunha, A. E. C., Cury, J. E. R. e Krogh, B. H. (2001). Supervision of event-driven hybrid systems: Modeling and synthesis, *in* Benedetto e Sangiovanni-Vincentelli (2001), pp. 247–260. Proceedings of the 4th International Workshop, HSCC 2001, Rome, Italy, March 28-30, 2001.

Grossman, R. L., Nerode, A., Ravn, A. P. e Rischel, H. (eds) (1993). *Hybrid Systems*, Vol. 736 of *Lecture Notes in Computer Science*, Springer-Verlag, New York.

He, K. e Lemmon, M. (1998). Modeling hybrid control systems using programmable timed petri nets, *European Journal of Automation, APII-JESA, Journal europeen des systemes automatises* **32**(9-10): 1187–1208.

Hedlund, S. (2003). *Computational Methods for Optimal Control of Hybrid Systems*, Ph.D. Thesis, Department of Automatic Control, Lund Institute of Technology, Sweden.

Henzinger, T. A. (1996). The theory of hybrid automata, *11th Annual IEEE Symposium on Logic in Computer Science (LICS)*, IEEE, IEEE Computer Society Press, pp. 278–292.

Henzinger, T. A. (2000). Masaccio: A formal model for embedded components, *in* J. van Leeuwen, O. Watanabe, M. Hagiya, P. D. Mosses e T. Ito (eds), *TCS 00: Theoretical Computer Science*, Vol. 1872 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 549–563.

Henzinger, T. A., Ho, P.-H. e Wong-Toi, H. (1997). HyTech: A Model Checker for Hybrid Systems, *Software Tools for Technology Transfer* **1**: 110–122.

Henzinger, T. A., Ho, P.-H. e Wong-Toi, H. (1998). Algorithmic analysis of nonlinear hybrid systems, *IEEE Transactions on Automatic Control, Special Issue on Hybrid Control Systems* **43**(4): 540–554.

Henzinger, T. A., Kopke, P. W., Puri, A. e Varaiya, P. (1995). What's decidable about hybrid automata?, *Proceedings of the 27th Annual Symposium on Theory of Computing*, ACM Press, pp. 373–382.

Henzinger, T. A., Kopke, P. W., Puri, A. e Varaiya, P. (1998). What's decidable about hybrid automata?, *Journal of Computer and System Sciences* **57**(1): 94–124.

Henzinger, T. A. e Majumdar, R. (2000). Symbolic model checking for rectangular hybrid systems, *Proceedings of the Sixth International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, Vol. 1785 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 142–156.

Henzinger, T. A. e Sastry, S. (eds) (1998). *Hybrid Systems: Computation and Control*, Vol. 1386 of *Lecture Notes in Computer Science*, Springer-Verlag. Proceedings of the First International Workshop, HSCC'98, Berkeley, California, USA, April 13-15, 1998.

Hespanha, J. (2004). Uniform stability of switched linear systems: Extensions of LaSalle's invariance principle, *IEEE Transactions on Automatic Control* **49**(4): 470–482.

Hopcroft, J. E., Motwani, R. e Ullmann, J. D. (2001). *Introduction to Automata Theory, Languages and Computation*, 2nd edn, Addison Wesley Publishing Company.

Horowitz, R. e Varaiya, P. (2000). Control design of an automated highway system, *Proceedings of the IEEE, Special Issue on Hybrid Systems: Theory and Applications* **88**(7): 913–925.

Ivancic, F. (2003). *Modeling and Analysis of Hybrid Systems*, Ph.D. Thesis, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA.

Johansson, K. H., Lygeros, J. e Sastry, S. (2004). *Modeling of hybrid systems*, EOLSS Publishers, chapter Chapter E6-43-28-01 in Theme 6.43, Control Systems, Encyclopedia of Life Support Systems (EOLSS), Developed under the auspices of UNESCO, Oxford, U.K.

Johansson, M. (1999). *Piecewise Linear Control Systems*, Ph.D. Thesis, Lund Institute of Technology, Gotemburg, Sweden.

Johnson, T. L. (1981). Analytic models of multistage processes, *Proceedings of the 20th IEEE Conference on Decision and Control – CDC*, San Diego, California.

Joseph, M. (ed.) (1988). *Formal Techniques in Real-Time and Fault-Tolerant Systems*, Vol. 331 of *Lecture Notes in Computer Science*, Springer-Verlag, Warwick, UK.

Jou, I.-C., Chang, C.-J. e Chen, H.-K. (1999). A hybrid neuro-fuzzy system for adaptive vehicle separation control, *J. VLSI Signal Process. Syst.* **21**(1): 15–29.

Julius, A. A. (2005). *On interconnection and equivalence of continuous and discrete systems: a behavioral perspective*, Ph.D. Thesis, Systems Signals and Control Group, Department of Applied Mathematics, University of Twente.

Juloski, A. L. (2004). *Observer Design and Identification Methods for Hybrid Systems: Theory and Experiments*, Ph.D. Thesis, Eindhoven Universtity of Technology, Eindhoven.

Kalman, R. E., Falb, P. L. e Arbib, M. A. (1969). *Topics in Mathematical System Theory*, McGraw-Hill, New York.

Kloas, M., Friesen, V. e Simons, M. (1995). Smile — A simulation environment for energy systems, *in* A. Sydow (ed.), *Proceedings of the 5th International IMACS-Symposium on Systems Analysis and Simulation (SAS'95)*, Vol. 18-19 of *Systems Analysis Modelling Simulation*, Gordon and Breach Publishers, pp. 503–506.

Koo, T. J. (2000). *Hybrid System Design and Embedded Controller Synthesis for Multi-Modal Control*, Ph.D. Thesis, Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, CA.

Koutsoukos, X. D. e Antsaklis, P. J. (1999). Hybrid control systems using timed petri nets: Supervisory control design based on invariant properties, *in* Antsaklis et al. (1999), pp. 142–162.

Koutsoukos, X. D., Antsaklis, P. J., Stiver, J. A. e Lemmon, M. D. (2000). Supervisory control of hybrid systems, *Proceedings of the IEEE, Special Issue on Hybrid Systems: Theory and Applications* **88**(7): 1026–1049.

Kowalewski, S., Stursberg, O., Fritz, M., Graf, H., Preußig, J., Simon, S., Stursberg, O. e Treseler, H. (1999). A case study in tool-aided analysis of discretely controlled continuous systems: The two-tanks-problem, in *Antsaklis et al. (eds.), Hybrid Systems V, LNCS vol. 1567* (Antsaklis et al., 1999), pp. 163–185.

Krogh, B. H. (1993). Condition/event signal interfaces for block diagram modeling and analysis of hybrid systems, *Proceedings of the 8th International Symposium on Intelligent Control* pp. 180–185.

Krogh, B. H. (2000). Hybrid systems: State of the art and perspectives, Plenária apresentada no XIII Congresso Brasileiro de Automática – CBA 2000. Florianópolis, SC.

Krogh, B. H. (2002). Recent advances in discrete analysis and control of hybrid systems, Invited talk, 6th International Workshop on Discrete Event Systems (WODES'02). Zaragoza, Spain.

Kuo, B. C. (1992). *Digital Control Systems*, 2nd edn, Saunders College Publishing.

Labinaz, G., Bayoumi, M. M. e Rudie, K. (1997). A survey of modeling and control of hybrid systems, *Annual Reviews in Control* **21**: 79–92.

Lafferriere, G., Pappas, G. e Yovine, S. (1999). A new class of decidable hybrid systems, *in* Vaandrager e van Schuppen (1999), pp. 137–151. Proceedings of the Second International Workshop, HSCC'99, Berg en Dal, The Netherlands, March 29-31, 1999.

Le Bail, J., Alla, H. e David, R. (1991). Hybrid petri nets, *Proceedings of the 1st International European Control Conference, ECC91*, Grenoble, France, pp. 1472–1477.

Leal, A. B. e Cury, J. E. R. (2002). Controle Modular de Sistemas Condição/Evento, *Anais do XIV Congresso Brasileiro de Automática – CBA2002*, Natal, RN, pp. 455–460.

Leal, A. B. e Cury, J. E. R. (2004a). Modular supervision of hybrid systems: A DES approach, *Proceedings of the 7th International Workshop on Discrete Event Systems – WODES'04*, Reims – France, pp. 169–174.

Leal, A. B. e Cury, J. E. R. (2004b). Modular supervisory control of event-driven hybrid systems, *Proceedings of the 2nd IFAC Symposium on System, Structure and Control – SSSC'04*, Oaxaca, Mexico, pp. 252–257.

Leal, A. B. e Cury, J. E. R. (2004c). On the existence of optimal solutions for the modular supervisory control of hybrid systems, *Proceedings of the 43rd IEEE Conference on Decision and Control – CDC'04*, Paradise Island, Bahamas, pp. 941–946.

Leal, A. B. e Cury, J. E. R. (2005). Supervisory control of hybrid systems: a modular approach. Submetido ao IEE Proceedings – Control Theory and Applications.

Lee, E. A. e Zheng, H. (2005). Operational semantics of hybrid systems, *in* Morari e Thiele (2005), pp. 9–11.

Lemmon, M. D., He, K. X. e Markovsky, I. (1999). Supervisory hybrid systems, *IEEE Control Systems* **19**(4): 42–55.

Lennartson, B., Tittus, M., Egardt, B. e Pettersson, S. (1996). Hybrid systems in process control, *IEEE Control Systems Magazine* **16**(5): 45–56.

Li, Z., Soh, C. B. e Xu, X. (2000). Lyapunov stability of a class of hybrid dynamic systems, *Automatica* **36**: 297–302.

Liberzon, D. (2000). Nonlinear stabilization by hybrid quantized feedback, *HSCC*, pp. 243–257.

Liberzon, D. e Morse, A. S. (1999). Basic problems in stability and design of switched systems, *IEEE Control Systems Magazine* **19**(5): 59–70.

Lima, A. M. N. e Miranda, M. V. C. (2004). Verification and control of hybrid systems, *Proceedings of the International Conference on Informatics, Simulation, Control and Modelling*, Vol. 1, pp. 1–6.

Livadas, C. (1997). *Formal Verification of Safety-Critical Hybrid Systems*, Master of engineering thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology.

Livadas, C., Lygeros, J. e Lynch, N. A. (2000). High-Level Modeling and Analysis of the Traffic Alert and Collision Avoidance System (TCAS), *Proceedings of the IEEE, Special Issue on Hybrid Systems: Theory and Applications* **88**(7): 926–948.

Livadas, C. e Lynch, N. A. (1998). Formal verification of safety-critical hybrid systems, *in* Henzinger e Sastry (1998), pp. 253–272. Proceedings of the First International Workshop, HSCC'98, Berkeley, California, USA, April 13-15, 1998.

Lygeros, J. (1996). *Hierarchical, Hybrid Control of Large Scale Systems*, Ph.D. Thesis, University of California at Berkeley, Berkeley, California.

Lygeros, J. (2004). Lecture notes on hybrid systems, Notes for an ENSIETA short course.

Lygeros, J., Godbole, D. N. e Sastry, S. S. (1998). Verified hybrid controllers for automated vehicles, *IEEE Transactions on Automatic Control, Special Issue on Hybrid Control Systems* **43**(4): 522–539.

Lygeros, J., Johansson, K. H., Simic, S. N., Zhang, J. e Sastry, S. (2003). Dynamical properties of hybrid automata, *IEEE Transactions on Automatic Control* **48**: 2–17.

Lygeros, J., Tomlin, C. e Sastry, S. (2002). Art of hybrid systems, Compendium of Lecture Notes for the Hybrid Systems Class.

Lynch, N. e Krogh, B. H. (eds) (2000). *Hybrid Systems: Computation and Control*, Vol. 1790 of *Lecture Notes in Computer Science*, Springer-Verlag. Proceedings of the Third International Workshop, HSCC 2000, Pittsburgh, PA, USA, March 23-25, 2000.

Lynch, N., Segala, R. e Vaandrager, F. (2003). Hybrid I/O Automata, *in Information and Computation*, Vol. 185, Elsevier Science, Academic Press, USA, pp. 105–157.

Lynch, N., Segala, R., Vaandrager, F. e Weinberg, H. B. (1996). Hybrid i/o automata, *in* Alur, Henzinger e Sontag (1996), pp. 496–510.

Maler, O. (ed.) (1997). *Hybrid and Real-Time Systems*, Vol. 1201 of *Lecture Notes in Computer Science*, Springer-Verlag. International Workshop, HART'97, Grenoble, France, March 26-28, 1997, Proceedings.

Maler, O., Manna, Z. e Pnueli, A. (1992). From timed to hybrid systems, *in* de Bakker et al. (1992), pp. 474–484. Proceedings of the REX Workshop, June 3-7, 1991, Mook, The Netherlands.

Manna, Z. e Pnueli, A. (1991). *The Temporal Logic of Reactive and Concurrent Systems*, Springer-Verlag.

Manna, Z. e Pnueli, A. (1993). Verifying hybrid systems, *in* Grossman et al. (1993), pp. 4–35.

Manna, Z. e Sipma, H. (1998). Deductive verification of hybrid systems using STeP, *in* Henzinger e Sastry (1998), pp. 305–318. Proceedings of the First International Workshop, HSCC'98, Berkeley, California, USA, April 13-15, 1998.

Mattsson, S. E., Otter, M. e Elmqvist, H. (1999). Modelica Hybrid Modeling and Efficient Simulation, *Proceedings of the 38th IEEE Conference on Decision and Control, CDC'99*, Phoenix, Arizona, USA, pp. 3502–3507.

Michel, A. (1999). Recent trends in the stability analysis of hybrid dynamical systems, *IEEE Transactions on Circuits and Systems I* **46**(1): 120–134.

Michel, A. e Hu, B. (1999). Towards a stability theory of general hybrid dynamical systems, in *Automatica* (Morse et al., 1999b), pp. 371–384.

Miller, J. S. (2000). Decidability and complexity results for timed automata and semi-linear hybrid automata, *in* Lynch e Krogh (2000), pp. 296–309. Proceedings of the Third International Workshop, HSCC 2000, Pittsburgh, PA, USA, March 23-25, 2000.

Miranda, M. V. C. (2003). *Contribuição ao uso de métodos formais no estudo de circuitos de eletrônica de potência*, Tese (doutorado), Programa de Pós-Graduação em Engenharia Elétrica - Universidade Federal de Campina Grande, Campina Grande, Paraíba.

Miranda, M. V. C. e Lima, A. M. N. (2003). Modelagem e verificação de propriedades de circuitos de eletrônica de potência, utilizando autômatos híbridos, *Anais do Simpósio Brasileiro de Automação Inteligente*, Vol. 1, Bauru, SP, pp. 953–958.

Moor, T., Davoren, J. M. e Raisch, J. (2001). Modular supervisory control of a class of hybrid systems in a behavioural framework, *Proc. European Control Conference ECC2001*, Porto, Portugal, pp. 870–875.

Moor, T., Davoren, J. M. e Raisch, J. (2002). Strategic refinements in abstraction based supervisory control of hybrid systems, *Proceedings of the 6th International Workshop on Discrete Event Systems – WODES'02*, Zaragoza, Spain, pp. 329–334.

Moor, T. e Raisch, J. (1999). Supervisory control of hybrid systems within a behavioural framework, *System and Control Letters, special issue on Hybrid Control Systems* **38**(3): 157–166.

Moor, T. e Raisch, J. (2002). Abstraction based supervisory controller synthesis for high order monotone continuous systems, *in* S. Engell, G. Frehse e E. Schnieder (eds), *Modelling, Analysis, and Design of Hybrid Systems*, Vol. 279 of *Lecture Notes in Control and Information Sciences*, Springer-Verlag, pp. 247–265.

Moor, T., Raisch, J. e Davoren, J. M. (2001). Computational advantages of a two-level hybrid control architecture, *Proceedings on the 40th IEEE Conference on Decision and Control*, Orlando, Florida, pp. 358–363.

Moor, T., Raisch, J. e O'Young, S. D. (1998). Supervisory control of hybrid systems via l-complete approximations, *WODES'98- International Workshop on Discrete Event Systems*, pp. 426–431.

Moor, T., Raisch, J. e O'Young, S. D. (2002). Discrete supervisory control of hybrid systems based on l-complete approximations, *Journal of Discrete Event Dynamic Systems* **12**(1): 83–107.

Moore, E. (1956). Gedanken experiments on sequential machines, *Automata Studies*, Princeton University Press, pp. 129–153.

Morari, M. e Thiele, L. (eds) (2005). *Hybrid Systems: Computation and Control, 8th International Workshop, HSCC 2005, Zurich, Switzerland, March 9-11, 2005, Proceedings*, Vol. 3414 of *Lecture Notes in Computer Science*, Springer.

Morse, A. S. (ed.) (1997). *Control Using Logic-Based Switching*, Vol. 222 of *Lecture Notes in Control and Information Sciences*, Springer-Verlag, New York.

Morse, A. S., Pantelides, C. C., Sastry, S. S. e Schumacher, J. M. (1999a). Introduction to the special issue on hybrid systems, *Automatica* **35**(3): 347–348. (Editorial).

Morse, A. S., Pantelides, C. C., Sastry, S. S. e Schumacher, J. M. (eds) (1999b). *Special Issue on Hybrid Systems*, Vol. 35, No. 3 of *Automatica*.

Mosterman, P. J. (1997). *Hybrid Dynamic Systems: A hybrid bond graph modeling paradigm and its application in diagnosis*, Ph.D. Thesis, Vanderbilt University, Nashville, US.

Mosterman, P. J. (1999). An overview of hybrid simulation phenomena and their support by simulation packages, *in* Vaandrager e van Schuppen (1999), pp. 165–177. Proceedings of the Second International Workshop, HSCC'99, Berg en Dal, The Netherlands, March 29-31, 1999.

Nerode, A. e Kohn, W. (1993). Models for hybrid systems: Automata, topologies, controllability, observability, *in* Grossman et al. (1993), pp. 317–356.

Nicollin, X., Olivero, A., Sifakis, J. e Yovine, S. (1993). An approach to the description and analysis of hybrid systems, *in* Grossman et al. (1993), pp. 149–178.

Niinomi, T. e Krogh, B. H. (1995). Modeling and analysis of switchedmode hybrid systems driven by threshold events, *in* H. Khalil, J. Chow e P. A. Ioannou (eds), *Proc. of Workshop on Advances in Control Systems and its Applications*, SpringerVerlag, pp. 155–172.

Niinomi, T., Krogh, B. H. e Cury, J. E. R. (1996). Refinements of approximating automata for synthesis of supervisory controllers for hybrid systems, *in* Alur, Henzinger e Sontag (1996), pp. 475–484.

Nikoukhah, R. e Steer, S. (1997). SCICOS A dynamic system builder and simulator user's guide - version 1.0, *Technical Report 0207*, INRIA, Rocquencourt, France.

Palomino Bean, S. (2002). *Abordagens LMI para análise de uma classe de sistemas híbridos*, Tese (doutorado), Programa de Pós Graduação em Engenharia Elétrica, Universidade Federal de Santa Catarina, Florianópolis, Brasil.

Pavlidis, T. (1967). Stability of systems described by differential equations containing impulses, *IEEE Transactions on Automatic Control* **12**(1): 43–57.

Peleties, P. A. (1992). *Modeling and Design of Interacting Continuous-Time/Discrete Event Systems*, Ph.D. Thesis, School of Electrical Engineering, Purdue University, West Lafayette, IN.

Peleties, P. e DeCarlo, R. (1988). Modeling of interacting continuous time and discrete event systems: An example, *Proc. of the 26th Annual Allerton Conference on Communication, Control and Computing* **2**: 1150–1159.

Peleties, P. e DeCarlo, R. (1989). A modeling strategy with event structures for hybrid systems, *Proc. 28th IEEE Conference on Decision and Control* **2**: 1308–1313.

Pepyne, D. L. e Cassandras, C. G. (2000). Optimal control of hybrid systems in manufacturing, *Proceedings of the IEEE, Special Issue on Hybrid Systems: Theory and Applications* **88**(7): 1108–1123.

Pettersson, S. (1999). *Analysis and Design of Hybrid Systems*, Ph.D. Thesis, Chalmers University of Technology, Gotemburg, Sweden.

Pettersson, S. e Lennartson, B. (1995). Hybrid modeling focused on hybrid Petri nets, *Proc. of the 2nd European Workshop on Real-Time and Hybrid Systems*, Grenoble, France, pp. 303–309.

Pettersson, S. e Lennartson, B. (1996). Stability and robustness for hybrid systems, *Proceedings of the 35th IEEE Conference on Decision and Control (CDC'96)* pp. 1202–1207.

Pettersson, S. e Lennartson, B. (2002). Hybrid system stability and robustness verification using linear matrix inequalities, *International Journal of Control* **75**(16): 1355–1355.

Pettersson, S. e Lennartson, B. (2003). Stability Analysis of Hybrid Systems - A Gear-Box Application, *Nonlinear and Hybrid Systems in Automotive Control, Springer-Verlag*, pp. 373–389.

Pinto, A., Sangiovanni-Vincentelli, A. L., Carloni, L. P. e Passerone, R. (2005). Interchange formats for hybrid systems: Review and proposal., *in* Morari e Thiele (2005), pp. 526–541.

Pnueli, A. e Sifakis, J. (eds) (1995). *Special Issue on Hybrid Systems*, Vol. 138, No. 1 of *Journal of Theoretical Computer Science*, Elsevier Science Publishers.

Potocnik, B., Bemporad, A., Torrisi, F. D., Music, G. e Zupancic, B. (2002). Scheduling of hybrid systems: Multi product batch plant, *Proceedings of the 2002 IFAC World Congress*, Barcelona.

Potoènik, B., Bemporad, A., Torrisi, F., Musiè, G. e Zupanèiè, B. (2003). Hysdel modeling and simulation of hybrid dynamical systems, *4rd IMACS Symposium on Mathematical Modelling – MATHMOD*, Vienna University of Technology, Vienna, Austria. Argesim Report, No. 24.

Puri, A. (1995). *Theory of Hybrid Systems and Discrete Event Systems*, Ph.D. Thesis, University of California, Berkeley.

Puri, A. e Varaiya, P. (1994). Decidability of hybrid systems with rectangular differential inclusions, *in* D. Dill (ed.), *Proceedings of the 6th Workshop on Computer-Aided Verification (CAV'94)*, Vol. 818 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 95–104.

Raisch, J., Itigin, A. e Moor, T. (2000). Hierarchical control of hybrid systems, *in* S. Engell, S. Kowalewski e J. Zaytoon (eds), *Proc. of the 4th International Conference on Automation of Mixed Processes: Dynamic Hybrid Systems ADPM2000 (Automatisation des Processus Mixtes: les Systemes Dynamiques Hybrides)*, Shaker-Verlag, Dortmund, Germany, pp. 67–72.

Raisch, J. e O'Young, S. D. (1998). Discrete approximation and supervisory control of continuous systems, *IEEE Transactions on Automatic Control, Special Issue on Hybrid Control Systems* **43**(4): 569–573.

Ramadge, P. J. (1990). On the periodicity of symbolic observations of piecewise smooth discrete-time systems, *IEEE Transactions on Automatic Control* **35**(7): 807–813.

Raymond, D. e Wood, D. (1995). GRAIL: A C++ Library for Automata and Expressions, *Journal of Symbolic Computation* **11**(1): 341–350.

Raymond, D. e Wood, D. (1996). The GRAIL Papers: Version 2.5, Computer Science Technical Report, HKUST-CS96-24.

Rodrigues, D. L. (2004). *Implementação de algoritmos para o controle supervisório modular de sistemas condição/evento*, Dissertação (mestrado), Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Santa Catarina, Florianópolis, SC.

Rozenvasser, E. N. (1967). General sensitivity equations of discontinuous systems, *Automat. Remote Control* pp. 400–404.

Senesky, M., Eirea, G. e Koo, T. J. (2003). Hybrid modelling and control of power electronics, *in* Wiedijk et al. (2003), pp. 450–465.

Shaikh, M. S. (2004). *Optimal Control of Hybrid Systems: Theory and Algorithms*, Ph.D. Thesis, Department of Electrical and Computer Engineering, McGill University, Montréal, Canada.

Silva, B. I., Richeson, K., Krogh, B. H. e Chutinan, A. (2000). Modeling and verifying hybrid dynamic systems using CheckMate, *in* S. Engell, S. Kowalewski e J. Zaytoon (eds), *Proc. of the 4th International Conference on Automation of Mixed Processes: Hybrid Dynamic Systems ADPM2000 (Automatisation des Processus Mixtes: les Systemes Dynamiques Hybrides)*, Shaker-Verlag, Dortmund, Germany, pp. 323–328.

Silva, B. I., Stursberg, O., Krogh, B. H. e Engell, S. (2001). An assessment of the current status of algorithmic approaches to the verification of hybrid systems, *Proceedings on the 40th IEEE Conference on Decision and Control*, Orlando, Florida, pp. 2867–2874.

Sipma, H. B. (1999). *Diagram-based verification of reactive, real-time, and hybrid systems*, Ph.D. Thesis, Department of Computer Science, Stanford University, Stanford, CA.

Song, M., Tarn, T.-J. e Xi, N. (2000). Integration of task scheduling, action planning and control in robotic manufacturing systems, *Proceedings of the IEEE, Special Issue on Hybrid Systems: Theory and Applications* **88**(7): 1097–1107.

Sreenivas, R. S. e Krogh, B. H. (1991). On condition/event systems with discrete state realizations, *Discrete Event Dynamic Systems: Theory and Applications* **1**: 209–236.

Stiver, J. A., Antsaklis, P. J. e Lemmom, M. D. (1995a). Hybrid control system design based on natural invariants, *Proceedings of the 34th IEEE Conference on Decision and Control*, New Orleans, LA, pp. 1455–1460.

Stiver, J. A., Antsaklis, P. J. e Lemmom, M. D. (1995b). Interface and controller design for hybrid control systems, *in* P. Antsaklis, W. Kohn, A. Nerode e S. Sastry (eds), *Hybrid Systems II*, Vol. 999 of *LNCS*, Springer-Verlag, pp. 462–492.

Stiver, J. A., Antsaklis, P. J. e Lemmon, M. D. (1996a). A Logical DES Approach to the Design of Hybrid Control Systems, *Mathematical and Computer Modeling – Special Issue on Discrete Event Systems* **23**(11/12): 55–76.

Stiver, J. A., Antsaklis, P. J. e Lemmon, M. D. (1996b). An invariant based approach to the design of hybrid control systems, *Proc. IFAC 13th Triennial World Congress*, Vol. J, San Francisco, CA, pp. 467–472.

Stursberg, O., Fehnker, A., Han, Z. e Krogh, B. (2004). Verification of a cruise control system using counterexample-guided search, *Control Engineering Practice* **12**(10): 1269–1278.

Tabuada, P. A. T. N. (2001). *Hierarchies and Compositional Abstractions of Hybrid Systems*, Ph.D. Thesis, Instituto Superior Técnico, Universidade Técnica de Lisboa, Lisboa.

Tavernini, L. (1987). Differential automata and their discrete simulators, *Nonlinear Analysis, Theory, Methods and Applications* **11**(6): 665–683.

Thevenon, L. (2000). *Représentation des Systèmes Hybrides Complexes par Flux de Données: Développement d'un Outil de Modélisation et de Simulation des Procédés Batch*, Thèse (doctorat), Institut National Polytechnique de Grenoble - INPG, Laboratoire d'Automatique, Grenoble, FR.

Thomas, J. (2004). *Estimation et Commande Prédictive à Horizon Glissant de Systèmes Hybrides*, Thèse (doctorat), Université de Paris-Sud.

Thompson, P. M. (1982). *Conic Sector Analysis of Hybrid Control Systems*, Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge.

Tiller, M. M. (2001). *Introduction to Physical Modeling with Modelica*, Vol. 615 of *The Kluwer International Series in Engineering and Computer Science*, 1st edn, Kluwer Academic Publishers, Boston.

Tittus, M. e Lennarston, B. (1998). Hierarchical supervisory control for batch processes, *Proceedings on the 37th IEEE Conference on Decision and Control*, Tampa, Florida, pp. 3251–3255.

Tomlin, C. J. (1998). *Hybrid Control of Air Traffic Management Systems*, Ph.D. Thesis, Department of Electrical Engineering and Computer Sciences, University of California at Berkeley.

Tomlin, C. J. e Greenstreet, M. R. (eds) (2002). *Hybrid Systems: Computation and Control*, Vol. 2289 of *Lecture Notes in Computer Science*, Springer-Verlag. Proceedings of the 5th International Workshop, HSCC 2002, Stanford, CA, USA, March 25-27, 2002.

Tomlin, C., Mitchell, I., Bayen, A. e Oishi, M. (2003). Computational techniques for the verification and control of hybrid systems, *Proceedings of the IEEE* **91**(7): 986–1001.

Tomlin, C., Pappas, G. e Sastry, S. (1998). Conflict resolution for air traffic management : A study in multi-agent hybrid systems, *IEEE Transactions on Automatic Control, Special Issue on Hybrid Control Systems* **43**(4): 509–521.

Torrico, C. R. C. (2003). *Controle Supervisório Hierárquico de Sistemas a Eventos Discretos: Uma Abordagem Baseada na Agregação de Estados*, Tese (doutorado), Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Santa Catarina, Florianópolis, SC.

Tsypkin, Y. Z. (1984). *Relay Control Systems*, Cambridge University Press, Cambridge.

Utkin, V. I. (1977). Variable structure systems with sliding modes: A survey, *IEEE Transactions on Automatic Control* **22**(2): 212–222.

Utkin, V. I. (1992). *Sliding Modes in Control Optimization*, 1st edn, Springer-Verlag, Berlin.

Vaandrager, F. W. e van Schuppen, J. H. (eds) (1999). *Hybrid Systems: Computation and Control*, Vol. 1569 of *Lecture Notes in Computer Science*, Springer-Verlag. Proceedings of the Second International Workshop, HSCC'99, Berg en Dal, The Netherlands, March 29-31, 1999.

van Beek, D. A., van den Ham, A. e Rooda, J. E. (2002). Modelling and control of process industry batch production systems, *Proceedings of the 15th Triennial World Congress of the International Federation of Automatic Control, CD-ROM*, Barcelona.

van Beek, D. e Rooda, J. (2000). Languages and Applications in Hybrid Modelling and Simulation: Positioning of CHI, *Control Engineering Practice* **8**(1): 81–91.

van der Schaft, A. e Schumacher, H. (2000). *An Introduction to Hybrid Dynamical Systems*, Vol. 251 of *Lecture Notes in Control and Information Sciences*, Springer-Verlag.

Varaiya, P. (1993). Smart cars on smart roads: Problems of control, *IEEE Transactions on Automatic Control* **38**(2): 195–207.

Villani, E. (2003). *Modelagem e análise de sistemas supervisórios híbridos*, Tese (doutorado), Escola Politécnica - Universidade de São Paulo, São Paulo.

Wiedijk, F., Maler, O. e Pnueli, A. (eds) (2003). *Hybrid Systems: Computation and Control, 6th International Workshop, HSCC 2003 Prague, Czech Republic, April 3-5, 2003, Proceedings*, Vol. 2623 of *Lecture Notes in Computer Science*, Springer.

Willems, J. C. (1991). Paradigms and puzzles in the theory of dynamical systems, *IEEE Transactions on Automatic Control* **36**: 259–294.

Wimpey, D. G. (1982). *Finite-State Control of Discrete-Time Continuous Processes: An Automata Motivated Approach*, Ph.D. Thesis, Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science, Cambridge.

Witsenhausen, H. S. (1966). A class of hybrid-state continuous-time dynamic systems, *IEEE Transactions on Automatic Control* **11**(2): 161–167.

Wonham, W. M. e Ramadge, P. J. (1987). On the supremal controllable sublanguage of a given language, *SIAM Journal of Control and Optimization* **25**(3): 637–659.

Wonham, W. M. e Ramadge, P. J. (1988). Modular supervisory control of discrete event systems, *Mathematics of Control, Signals, and Systems (MCSS)* **1**(1): 13–30.

Xie, D., Wang, L., Hao, F. e Xie, G. (2003). An lmi approach to disturbance rejection of switched systems, *International Journal of Hybrid Systems* **3**(3): 251–262.

Xie, G. e Wang, L. (2005). Controllability implies stabilizability for discrete-time switched linear systems., *in* Morari e Thiele (2005), pp. 667–682.

Ye, H., Michel, A. N. e Hou, L. (1998). Stability theory for hybrid dynamical systems, *IEEE Transactions on Automatic Control, Special Issue on Hybrid Control Systems* **43**(4): 461–474.

Young, K. D. e Özgüner, Ü. (eds) (1999). *Variable Structure Systems, Sliding Mode and Nonlinear Control*, Vol. 247 of *Lecture Notes in Control and Information Sciences*, Springer-Verlag, New York.

Zad, S. H. (1999). *Fault Diagnosis in Discrete-Event and Hybrid Systems*, Ph.D. Thesis, Systems Control Group, Department of Electrical & Computer Engineering, University of Toronto, Toronto, Canada.